

Approximation Bounds For Minimum Degree Matching

Bert Besser*

Institut für Informatik, Goethe-Universität Frankfurt am Main, Germany

Abstract. We consider the MINGREEDY strategy for Maximum Cardinality Matching. MINGREEDY repeatedly selects an edge incident with a node of minimum degree. For graphs of degree at most Δ we show that MINGREEDY achieves approximation ratio at least $\frac{\Delta-1}{2\Delta-3}$ in the worst case and that this performance is optimal among adaptive priority algorithms in the vertex model, which include many prominent greedy matching heuristics.

Even when considering expected approximation ratios of randomized greedy strategies, no better worst case bounds are known for graphs of small degrees.

Keywords: matching, greedy, approximation, priority algorithm.

1 Introduction

In the **Maximum Cardinality Matching** Problem a node disjoint subset of edges of maximum size is to be determined. Matching problems have many applications, e.g. image feature matching in computer vision or protein structure comparison in computational biology.

A maximum matching can be found in polynomial time, e.g. by the algorithm of Micali and Vazirani [MV80] running in time $O(\sqrt{|V|} \cdot |E| \cdot \alpha(|E|, |V|))$ where $\alpha(|E|, |V|)$ is the inverse Ackermann function [Vaz12]. The algorithm of Mucha and Sankowski [MS04] runs asymptotically faster on dense graphs, its runtime is $O(|V|^\omega)$ where $\omega < 2.38$ is the exponent needed to perform matrix multiplication.

However, there are much faster greedy algorithms that in practice compute very large matchings, even near optimal ones. Very good approximate solutions may already be satisfactory in some applications. If maximum matchings are needed, one can save lots of runtime when feeding large greedy matchings into optimal algorithms which iteratively improve an initial solution.

MINGREEDY. The (randomized) MINGREEDY algorithm computes a matching M by repeatedly picking edges incident to nodes of currently minimum degree, see Figure 1. MINGREEDY can be implemented in linear time $O(|V| + |E|)$.

The experimentally observed approximation performance is quite impressive. Tinhofer [Tin84] observed that on random graphs of varying density MINGREEDY performed superior to GREEDY (which randomly selects an edge) and to MRG (which randomly selects a node and subsequently an incident edge). In experiments of Frieze et al. [FRS93] on random cubic graphs MINGREEDY left only about 10 out of 10^6 nodes unmatched. On random graphs of small constant

* Partially supported by DFG SCHN 503/6-1.

```

M = ∅
repeat until all edges removed from input graph:
    select (random) node u of minimum non-zero degree
    select (random) neighbor v of u
    pick edge {u, v}, i.e. set M = M ∪ {{u, v}}
    remove all edges incident with u and v from input graph
return M

```

Fig. 1. The (randomized) MINGREEDY algorithm

average degree Magun [Mag97] observed that MINGREEDY produces extremely few “lost edges” in comparison with an optimal solution.

In an involved argument, Frieze et al. [FRS93] showed that $c_1 \cdot n^{\frac{1}{5}} \leq \lambda_n \leq c_2 \cdot n^{\frac{1}{5}} \cdot \ln(n)$ holds (c_1, c_2 being constants) for the expected number λ_n of nodes not being matched by MINGREEDY on random n -node cubic graphs.

Whereas the performance of MINGREEDY on random instances is very good, its worst case performance is poor. Poloczek [Pol12] constructed hard input instances on which MINGREEDY (and common variations of the algorithm) achieves approximation ratio at most $\frac{1}{2} + o(1)$ w.h.p.

MRG, GREEDY & SHUFFLE. The Modified Random Greedy algorithm, abbreviated MRG in literature, ignores node degrees and repeatedly selects a node and then a neighbor uniformly at random. The expected approximation ratio was shown to be at least $\frac{1}{2} + \frac{1}{400,000}$ by Aronson et al. [ADFS95].

The random edge algorithm GREEDY repeatedly selects an edge uniformly at random. For graphs with degrees bounded by Δ an expected lower bound on the approximation ratio of $\frac{\Delta}{2\Delta-1}$ was shown by Dyer and Frieze [DF91] and later improved by Miller and Pritikin [MP97] to $\frac{1}{2}(\sqrt{(\Delta-1)^2+1}-\Delta+2)$. If GREEDY prefers edges of degree-1 nodes, the KARP-SIPSER algorithm is obtained, which is asymptotically optimal w.h.p. on large sparse random graphs [KS81].

The SHUFFLE algorithm, proposed by Goel and Tripathi [GT12], is an adaptation of the RANKING algorithm of Karp et al. [KVV90] to non-bipartite graphs. SHUFFLE selects a random permutation π of the nodes and repeatedly matches the, according to π , first non-isolated node to its first unmatched neighbor. Chan et al. [CCWZ14] showed that SHUFFLE achieves an approximation ratio of at least $2 \cdot (5 - \sqrt{7})/9 \approx 0.523$.

Inapproximability. To show performance bounds for greedy algorithms, Borodin et al. [BNR02] proposed the model of *adaptive priority algorithms*. The model formalizes the greedy nature of an algorithm: while gathering knowledge about the input, irrevocable decisions have to be made to construct a solution.

Davis and Impagliazzo [DI04] introduced the *vertex model* to study adaptive priority algorithms for graph problems. Adaptive priority algorithms in the vertex model for the matching problem, which we call \mathcal{APV} -algorithms, implement powerful node and edge selection routines. In particular, in each step a node v and an incident edge is not picked arbitrarily but based on all knowledge already gathered about v and its neighbors, e.g. is a neighbor matched or unmatched,

what is the degree of a neighbor, what are the neighbors of a neighbor, etc. \mathcal{APV} -algorithms include GREEDY, KARPSIPSER, MRG, MINGREEDY, SHUFFLE and all vertex iterative algorithms, a class of algorithms defined in [GT12] as a generalization of SHUFFLE.

Despite the strength of \mathcal{APV} -algorithms, Poloczek [Pol12] constructed rather simple graphs with worst case approximation ratio at most $\frac{2}{3}$. (He also showed an inapproximability bound of $\frac{5}{6}$ for randomized priority algorithms.) Presented in the same thesis, for graphs with arbitrarily large degree Besser and Poloczek showed that no \mathcal{APV} -algorithm achieves worst case approximation ratio better than $\frac{1}{2} + o(1)$.

Contributions. From now on we reserve the name MINGREEDY for the deterministic version of MINGREEDY in which a node of minimum degree and an incident edge is picked by a worst case adversary.

We show that MINGREEDY approximates an optimal matching within a factor of $\frac{\Delta-1}{2\Delta-3}$ for graphs in which degrees are bounded by at most Δ . In the proof we analyze a variant of MINGREEDY which is also related with the KARPSIPSER algorithm.

We also show that the worst case approximation performance of MINGREEDY is optimal for (deterministic) \mathcal{APV} -algorithms. In particular, we improve the construction of Besser and Poloczek given in [Pol12] and present hard input instances of degree at most Δ for which any \mathcal{APV} -algorithm computes a matching of size at most $\frac{\Delta-1}{2\Delta-3} + o(1)$ times optimal.

Our worst case performance guarantees are stronger than the best known bounds on the expected performance of MRG and SHUFFLE ($\frac{1}{2} + \frac{1}{400.000}$ resp. ≈ 0.523), for small Δ , and of GREEDY (≈ 0.618 for $\Delta=3$), for all Δ .

Techniques. For our performance guarantees for MINGREEDY we study the matching graph composed of the edges of a matching M , computed by MINGREEDY, and of a maximum matching M^* . The connected components are alternating paths and cycles. Only paths of length three have poor “local” approximation ratio (of M -edges to M^* -edges). To obtain a global performance guarantee, we balance local approximation ratios by transferring “ M -funds” from rich to poor components using edges of the input graph.

Incorporating the properties of MINGREEDY within an amortized analysis is our technical contribution.

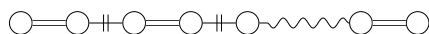
Overview. In Section 2 we present the charging scheme used to prove the performance guarantees for MINGREEDY. In Section 3 we show our $\frac{2}{3}$ bound for graphs of degree at most $\Delta = 3$. For graphs of degree at most $\Delta \geq 4$ we present in Section 4 our performance guarantee of $\frac{\Delta-1}{2\Delta-3}$. Our inapproximability results for \mathcal{APV} -algorithms are given in Section 5.

2 The Charging Scheme

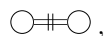
The Matching Graph. Let $G = (V, E)$ be a connected graph, M^* a maximum matching in G and M a matching computed by MINGREEDY when applied to input G . To analyze the worst case approximation ratio of MINGREEDY we investigate the graph

$$H = (V, M \cup M^*).$$

The connected components of H are paths and cycles composed of edges of M and M^* . For example, H contains so-called *(M-)augmenting paths*, or *paths* for short: an augmenting path X has m_X edges of M and $m_X^* = m_X + 1$ edges of M^* and starts and ends with an M^* -edge:



We call the two nodes of a path X which do not have an incident M -edge the *endpoints* of X (the leftmost and the rightmost node in the figure). Other connected components of H are edges of $M \cap M^*$



which we call *singletons*. For a singleton X we have $m_X = m_X^* = 1$. We may focus on these two component types:

Lemma 1. *There is a maximum matching M^* in G such that each connected component of H is a singleton or an augmenting path.*

Proof. Let M' be a maximum matching in $G = (V, E)$ and let MINGREEDY compute the matching M . We show how to transform M' into M^* . The connected components of the graph $(V, M \cup M')$ are singletons and alternating paths and cycles, where a path starts and ends with an M -edge or M' -edge and a cycle does not have path endpoints. To prove the statement we eliminate paths starting and ending both with an M -edge, paths starting and ending with different types of edges, and cycles. There is no path X of the first type, since if there was, then we could replace the m_X' many M' -edges of X with the $m_X' + 1$ many M -edges of X to obtain a matching larger than M' . In a component X of the latter types we replace the M' -edges of X with the M -edges of X : component X is replaced by m_X many singletons. \square

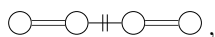
Local Approximation Ratios. To bound the approximation ratio

$$\alpha = |M|/|M^*|$$

of MINGREEDY, our approach is to bound *local approximation ratios*

$$\alpha_X = m_X/m_X^*,$$

of components X of H . Observe that a $\frac{1}{2}$ -path X



i.e. an augmenting path with $m_X = 1$ edge of M and $m_X^* = 2$ edges of M^* , has local approximation ratio $\alpha_X = \frac{1}{2}$, while all other components have local approximation ratios at least $\frac{2}{3}$. In particular, a singleton X has optimal local approximation ratio $\alpha_X = 1$. Thus we have to balance local approximation ratios. For any component X , we say that X has M -funds m_X and we introduce a change t_X to the M -funds of X such that the changed local approximation ratio of X is lower bounded by

$$\alpha_X = \frac{m_X + t_X}{m_X^*} \stackrel{!}{\geq} \beta$$

for appropriately chosen β . If $\sum_X t_X = 0$ holds, then the total M -funds $\sum_X m_X + t_X = \sum_X m_X = |M|$ are unchanged and hence MINGREEDY achieves approximation ratio at least

$$\alpha = |M|/|M^*| = \left(\sum_X m_X + t_X \right) / |M^*| \geq \left(\sum_X \beta m_X^* \right) / |M^*| = \beta.$$

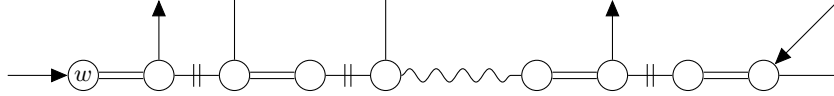


Fig. 2. An augmenting path with transfers to its path endpoints and from its M -covered nodes. Not all F -edges are used to move M -funds.

Transferring M -Funds. The idea is to transfer M -funds from rich components to poor components: as an example, we could transfer M -funds from a singleton to a $\frac{1}{2}$ -path. But which components should be involved in a transfer?

Observe that an augmenting path endpoint w is detrimental to its augmenting path X , since w decreases the local approximation ratio of X . So a transfer should push M -funds towards poor w from a rich M -covered node.

Since the G -neighbors of w are M -covered (otherwise M would not be maximal), our approach is to move M -funds to w from G -neighbors of w which belong to other components: M -funds are moved over certain edges in

$$F := E \setminus (M \cup M^*)$$

which connect the components of H , see Figure 2 for an illustration. We verify that a poor component, in particular a $\frac{1}{2}$ -path, is able to receive M -funds:

Lemma 2. *An augmenting path endpoint w is incident with F -edge, since the degree of w in G is at least*

$$d_G(w) \geq 2.$$

Proof. When MINGREEDY picks the first M -edge of the augmenting path of w , a node of degree at least two is selected (with an incident M -edge and M^* -edge). So w also has degree at least two and is incident not only with its M^* -edge. \square

When transferring M -funds over all F -edges we face the danger of augmenting path endpoints having large degree and drawing very large amounts of M -funds: rich components might become poor and now have too small local approximation ratio. The following definition prevents high degree path endpoints from wasting M -funds.

Definition 1. *Let w be an augmenting path endpoint. Edge $\{v, w\} \in F$ is a transfer if in the step of MINGREEDY matching v the degree of w drops to at most $d(w) \leq 1$. A transfer moves one coin with value θ from the component of v to the component of w .*

We frequently denote a transfer $\{v, w\}$ as (v, w) to stress its direction. In order to refer to transfers from/to a given component X , we also call (v, w) a *debit from v* and a *credit to w* . I.e., debits from a component are transfers directed from its M -covered nodes, credits to an augmenting path are transfers directed to its path endpoints.

Bounding Local Approximation Ratios. Let X be a component of H and denote the numbers of debits and credits of nodes of X by d_X respectively c_X . We call $c_X - d_X$ the *balance* of X .¹ Given an upper bound on the debits from X and a lower bound on the credits to X , we obtain a bound on the balance of X of at least

$$c_X - d_X \geq T_X.$$

Whenever we move M -funds over an edge $\{v, w\} \in F$, we transfer an amount θ of M -funds to the augmenting path endpoint w using one coin. Hence the local approximation ratio of X is at least

$$\alpha_X = \frac{m_X + \theta c_X - \theta d_X}{m_X^*} \geq \frac{m_X + \theta T_X}{m_X^*}.$$

In the analysis we find T_X and θ such that $\alpha_X \geq \beta$ for all components. Hence MINGREEDY computes matchings of size at least β times optimal.

¹ Previous versions of this work use the notion of *total debits* of a component X , which are defined as $d_X - c_X$. This notion is equivalent to the notion of the balance of X .

2.1 Balance Bounds

For each path X we demand the following: both nodes of each M -edge $\{x, x'\}$ of X pay at most

$$d_{\{x, x'\}} \leq 2(\Delta - 2) \quad (-\text{Edge})$$

coins and X receives at least

$$c_X \geq 2 \quad (+\text{Endpoints})$$

coins. Therefore, the minimum balance of path X is $2 - D_X$ for $D_X := 2m_X(\Delta - 2)$. In particular, bounds $(-\text{Edge})$ and $(+\text{Endpoints})$ are bounds on numbers of coins and not on numbers of transfers: we demand that both hold w.r.t. transfers as well as *donations*, the latter of which are another way to move coins that we introduce later.

To show that all components have local approximation ratio at least $\frac{\Delta-1}{2\Delta-3}$ we will verify balance bounds

$$\begin{aligned} -d_X &\geq -2(\Delta - 1) + 2 && \text{for any singleton } X \text{ and } (\pm\text{Singleton}) \\ c_X - d_X &\geq 2 - D_X + 2(\Delta - 2) && \text{for any path } X. \quad (\pm\text{Path}) \end{aligned}$$

Lemma 3. *If balance bounds $(\pm\text{Singleton})$ and $(\pm\text{Path})$ hold then all components have local approximation ratio at least $\frac{\Delta-1}{2\Delta-3}$.*

Proof. Choose $\theta = \frac{1}{2(2\Delta-3)}$. Using $(\pm\text{Singleton})$, the local approximation ratio of a singleton X is at least $\frac{1+\theta(c_X-d_X)}{1} \geq 1 - 2\theta(\Delta - 2)$, which in turn equals

$$1 - 2\theta(\Delta - 2) = 1 - \frac{2(\Delta - 2)}{2(2\Delta - 3)} = \frac{\Delta - 1}{2\Delta - 3}. \quad (1)$$

Therefore each singleton has local approximation ratio at least $\frac{\Delta-1}{2\Delta-3}$.

Using $(\pm\text{Path})$, we get that the local approximation ratio of a path X is at least

$$\begin{aligned} \frac{m_X + \theta(c_X - d_X)}{m_X^*} &\geq \frac{m_X - 2\theta m_X(\Delta - 2) + 2\kappa(\Delta - 1)}{m_X + 1} \\ &= 1 - 2\theta(\Delta - 2) + \frac{2\theta(2\Delta - 3) - 1}{m_X + 1} \\ &= 1 - 2\theta(\Delta - 2), \end{aligned}$$

which by (1) is bounded by at least $\frac{\Delta-1}{2\Delta-3}$ as well, i.e. all paths have local approximation ratio at least $\frac{\Delta-1}{2\Delta-3}$. \square

2.2 Organisation of the Proof of (\pm Singleton) and (\pm Path)

Consider the 1-2-MINGREEDY algorithm given in Figure 3. Observe that a sequence of edges picked by the (deterministic) MINGREEDY algorithm is a valid sequence of edges to be picked by 1-2-MINGREEDY. Therefore 1-2-MINGREEDY achieves no better worst case approximation ratio than MINGREEDY and obtaining performance guarantees for 1-2-MINGREEDY implies the same approximation performance for MINGREEDY.

Note that 1-2-MINGREEDY can also be understood as a refined variant of the (deterministic) KARSIPSE algorithm, which selects an arbitrary edge whenever all degrees are at least two and otherwise selects an arbitrary node of minimum degree, i.e. of degree one, and matches it with an arbitrary neighbor.

```

M = ∅
repeat until all edges removed from input graph:
  if each node has degree at least 3:
    select (arbitrary) edge {u, v}
  else:
    select (arbitrary) node u of minimum non-zero degree
    select (arbitrary) neighbor v of u
  pick edge {u, v}, i.e. set M = M ∪ {{u, v}}
  remove all edges incident with u and v from input graph
return M

```

Fig. 3. The (deterministic) 1-2-MINGREEDY algorithm

We prepare the proof with some basic observations in Section 2.3. Then, for 1-2-MINGREEDY we verify bounds ($-$ Edge), ($+$ Endpoints), (\pm Singleton), and (\pm Path) for graphs of maximum degree $\Delta = 3$ in Section 3, where our argument will rely on transfers only. In Section 4 we analyze graphs of maximum degree $\Delta \geq 4$, where we will introduce *donations* as an additional way to move coins.

2.3 Preparations

Since transfers move coins from M -covered nodes over F -edges to adjacent path endpoints, the following bounds on the number of outgoing and incoming transfers are an immediate consequence.

In particular, the following bounds imply ($-$ Edge) as well as ($+$ Endpoints) in case coins are moved only in transfers but not in donations.

Lemma 4. *Let an M -edge $\{u, v\}$ be given.*

If $\{u, v\}$ is the edge of a singleton, then both nodes pay at most $2(\Delta - 1)$ debits.

If $\{u, v\}$ is a path M -edge, then both nodes pay at most $2(\Delta - 2)$ debits.

Proof. Since only F -edges can be transfers, it suffices to bound the number of F -edges incident with each of u and v . Each M -covered node of a singleton is incident with at most $\Delta - 1$ many F -edges. Each M -covered node of a path is incident with at most $\Delta - 2$ many F -edges. \square

Lemma 5. *Let w be an endpoint of path X . Endpoint w receives at least one credit. Path X receives at least $c_X \geq 2$ credits.*

Proof. Let $d_G(w)$ denote the degree of w in G and denote the current degree of w by $d(w)$. By Lemma 2 we have $d_G(w) \geq 2$.

Since w is a path endpoint, node w never gets matched. Consider the step when the degree of w drops from $d(w)$ to 0, where $d(w) \in \{1, 2\}$.

If $d(w) = 2$ holds, then an F -edge incident with w gets removed and we obtain $c_w \geq 1$.

If $d(w) = 1$ holds, then either an F -edge incident with w gets removed and we have $c_w \geq 1$, or the M^* -edge of w gets removed. In the latter case the degree of w dropped from $d_G(w) \geq 2$ to $d(w) = 1$ when an F -edge incident with w was removed, and again we have $c_w \geq 1$.

Finally, observe that since path X has two endpoints, path X receives at least $c_X \geq 2$ credits. \square

3 Maximum Degree $\Delta = 3$

In this section we show

Theorem 1. *The 1-2-MINGREEDY algorithm achieves approximation ratio at least $\frac{\Delta-1}{2\Delta-3} = \frac{2}{3}$ for graphs of maximum degree $\Delta = 3$.*

In the proof we verify $(\pm\text{Singleton})$ and $(\pm\text{Path})$, which for $\Delta = 3$ reduce to

$$\begin{array}{lll} -d_X \geq -2 & \text{for each singleton } X \text{ and} & (\pm\text{Singleton}^3) \\ c_X - d_X \geq 2 - 2m_X + 2 & \text{for each path } X. & (\pm\text{Path}^3) \end{array}$$

Coins will be moved using transfers only. Therefore $(-\text{Edge})$ and $(+\text{Endpoints})$ hold: in case only transfers are used, we have already verified both bounds in Lemmas 4 and 5.

3.1 Two Debits are Missing

Recall that the minimum balance of a singleton X is $-2(\Delta - 1) = -4$, i.e. singleton X pays at most four debits. Our plan to verify $(\pm\text{Singleton}^3)$ is to show that X pays two debits less than maximum: we say that two debits are *missing*.

Similarly, the minimum balance of a path X is $2 - D_X = 2 - 2m_X$, where D_X is the maximum number of debits paid by X . Again, if X has two missing debits then X is balanced as required in $(\pm\text{Path}^3)$. However, it might be the case that X has less than two missing debits. In this case the balance of X will be increased by additional credits.

Organization of the Proof. We start with the proof of two missing debits for singletons in Lemma 6 (in Lemma 6 c), in particular); this result concludes the proof of $(\pm\text{Singleton}^3)$.

More generally, Lemma 6 also applies to paths and identifies those cases in which paths have two missing debits as well. Lemma 6 also prepares the analysis of the special case in which a path X has less than two missing debits: we will argue that X has exactly one missing debit and in Lemma 7 in Section 3.2 we prove that the extra debit is compensated by an additional credit to X . Hence Lemmas 6 and 7 conclude the proof of $(\pm\text{Path}^3)$.

Lemma 6. *Assume that degrees are bounded by at most $\Delta = 3$. Consider the creation step of component X , and assume that the 1-2-MINGREEDY algorithm selects node u with current degree $d(u)$. Component X has two missing debits if*

- a) *we have $d(u) = 1$*
- b) *we have $d(u) = \Delta = 3$*
- c) *component X is a singleton.*

Proof. Let X be created in step s when u is matched with, say, node v . If neither u nor v pays a debit, then two debits are missing and we are done. Hence in the rest of the proof we may assume that at least one of u and v pays a debit, which we denote by (x, w) for $x \in \{u, v\}$.

We prove a). If $d(u) = 1$ holds at step s , then no F -edges are incident with u . Hence u has two missing debits by Definition 1.

We prove b). Since (x, w) is a transfer, step s removes edge $\{x, w\}$ from G and after step s the degree of w is $d'(w) \leq 1$. Observe that in step s at most two edges incident with w are removed. Since the degree of w is at least $d(w) \geq d(u) = 3$ before step s , we have $d'(w) = 1$ at step $s + 1$. But since endpoint w is never selected, another degree-1 node $y \neq w$ is selected next in step $s + 1$. Observe that the degree of y also drops from $d(y) = 3$ to $d'(y) = 1$ in step s , namely when incident edges $\{u, y\}$ and $\{v, y\}$ are removed.

- If **y belongs to a component $Y \neq X$** other than X , then both $\{u, y\}$ and $\{v, y\}$ are F -edges. But $\{u, y\}$ and $\{v, y\}$ are not debits, since u, v and y are M -covered. Hence we have found a missing debit for each of u and v .
- If **y is a node of X** (note that in this case X must be a path), then nodes u, v , and y form a triangle and one of $\{u, y\}$ and $\{v, y\}$, say $\{u, y\}$, is an F -edge. Since both u and y are M -covered, edge $\{u, y\}$ is not a transfer and hence both u and y have a missing debit.

We prove c). We may assume that $d(u) = 2$ holds at creation of singleton X , since a) and b) apply in particular to singletons. Observe that a debit is missing from node u . In order to obtain a contradiction, we assume that this is the only missing debit from X . Consequently, node u pays exactly one debit, say to w_u , and v pays exactly two debits, say to w_v and w'_v .

Since at creation of X node u has degree $d(u) = 2$ and is adjacent to v , node u is adjacent to at most one of w_v and w'_v . Since edges $\{u, w_u\}$, $\{v, w_v\}$, and $\{v, w'_v\}$ are transfers, Definition 1 implies that the degrees of w_u , w_v and, w'_v are at most 1 after step s . We distinguish two cases.

- If **u is adjacent to neither w_v nor w'_v** , then w_u , w_v , and w'_v have degree exactly 1 after step s : their degrees were at least the minimum degree of $d(u) = 2$ before $\{u, v\}$ was picked, and their degrees dropped to at most one afterwards.
- Now consider the case that **u is adjacent to w_v or w'_v** , say $w_u = w_v$ holds. Then the degree of w'_v drops by at most one when edge $\{u, v\}$ is picked, and hence w'_v has degree exactly 1 afterwards.

In both cases, no other degrees than those of w_u , w_v , and w'_v dropped in step s . Since in step $s+1$ one of these endpoints has degree exactly one, one of w_u , w_v , or w'_v is selected and matched next. A contradiction, since path endpoints are never matched by 1-2-MINGREEDY. \square

3.2 One Missing Debit and an Additional Credit

To finish the proof of Theorem 1 it remains to verify $(\pm\text{Path}^3)$ for any given path X . Recall that the balance of X is at least $c_X - d_X \geq 2 - D_X = 2 - 2m_X$. To prove a balance of at least $c_X - d_X \stackrel{!}{\geq} 2 - 2m_X + 2$ it suffices to show that two debits are missing for X .

Since by Lemma 2 path X is created in a step selecting a node u of current degree $d(u) \geq 2$, we analyze the cases that in the creation step of X we have $d(u) = 2$ or $d(u) = 3$.

In the latter case, two debits are missing by Lemma 6 b). Hence, for the rest of the proof we may assume that $d(u) = 2$ holds. Observe that no F -edge is incident with u when being selected, hence a debit is missing for u .

If an additional debit is missing for X , then we are done once again. So assume from here on that the *only* missing debit for X is missing for u . Then $d_X = D_X - 1 = 2m_X - 1$ holds. Rather than proving a contradiction, in Lemma 7 we show that X receives at least *three* credits: we obtain $c_X - d_X \geq 3 - 2m_X + 1$, as required in $(\pm\text{Path}^3)$.

Lemma 7. *Assume that degrees are bounded by at most $\Delta = 3$. Let X be a path which pays $d_X = 2m_X - 1$ debits. Then X receives at least $c_X \geq 3$ credits.*

Proof. We assume that $c_X < 3$ holds and show a contradiction.

By assumption and as a consequence of Lemma 5, each endpoint of X receives exactly one credit. Moreover, we have already argued before Lemma 7 that the node u selected to create X has degree $d(u) = 2$ at creation, and that u has the only missing debit of X . Consequently, all other M -covered nodes of X have exactly one debit, since $d_X = 2m_X - 1$ holds. Let u be matched with node v .

First we consider the case that X is a $\frac{1}{2}$ -path with $m_X = 1$. Let w_u and w_v be the endpoints of X such that we have $\{u, w_u\}, \{v, w_v\} \in M^*$. Note that this implies $w_u \neq w_v$. Node v pays the only debit, say to the path endpoint y . Observe that in the creation step of X the degrees of w_u, w_v , and y all drop, and no other degrees drop.

If the degree of w_v is still at least two after X was created, then w_v receives two credits over two still incident F -edges. A contradiction to our assumption that $c_X < 3$ holds.

Hence we may assume from here on that after creation of X the degree of w_v is at most $d'(w) \leq 1$. But when $\{u, v\}$ is picked we have $d(u) = 2$, i.e. endpoint w_v is not incident with u and the degree of w_v drops by at most one. Consequently, endpoint w_v has degree exactly $d'(w) = 1$ after creation, which is the new minimum degree in the graph. But since only the degrees of w_u, w_v , and y dropped in the creation step, one of w_u, w_v and y is matched in the next step. A contradiction is obtained since path endpoints are never matched. The possible configurations are depicted in Figure 4.

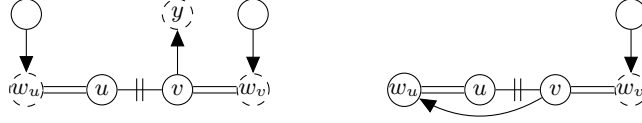


Fig. 4. Creating a $\frac{1}{2}$ -path which pays one debit and receives exactly two credits: dashed endpoints have degree one after u and v are matched (not all edges are drawn)

Now consider the case that path X has $m_X \geq 2$ edges of M . After edge $\{u, v\}$ is picked in the creation step of X , at least one path endpoint w of X is still connected with its unique M^* -neighbor, call it x . We consider the step when x becomes matched, say with its M -neighbor x' . In this step, we denote the current degree of a node z as $d(z)$.

Recall that x pays a debit, say to endpoint y . Since w and y are endpoints and are thus never matched, both w and y are not yet isolated. Thus node x is still connected with x', w , and y and has degree at least $d(x) \geq 3$.

Also, we may assume that w has degree at most $d(w) \leq 2$. To see this, assume that the degree of w is still $\Delta = 3$ and observe that the two F -edges incident with w become credits, which contradicts our assumption that $c_X < 3$ holds.

Moreover, we argue that the degree of w is exactly $d(w) = 2$. Why? Recall that u has the only missing debit from X . Hence x' pays a debit, i.e. node x' is adjacent to an endpoint of a path. Since x' is also adjacent to x , node x' has degree at least $d(x') \geq 2$. But the degree of x' is not larger than the degree $d(w)$ of w , i.e. it is at most $d(x') \leq d(w) \leq 2$. Therefore the degree of x' is exactly $d(x') = 2$. Consequently, endpoint w has degree exactly $d(w) = 2$ as well.

All still present neighbors of x and x' are endpoints of paths, and only their degrees drop when edge $\{x', x\}$ is picked. The possible configurations are depicted in Figure 5. We distinguish two cases.

- If x' is not adjacent to w , then the degree of w drops by exactly one to exactly (the three leftmost configurations in Figure 5). Hence either w or another path endpoint is selected in the next step. A contradiction, since a path endpoint is never matched.
- Lastly, assume that x' and w are adjacent (the rightmost configuration in Figure 5). Then w becomes isolated in that step. We consider the recipient y of the debit from x . Since x' is adjacent to w and x and has degree exactly $d(x') = 2$, endpoint y is not adjacent to x' . Therefore the degree of y drops by exactly one. Since the degree of y drops from at least $d(y) \geq 2$ to at most 1, by Definition 1 we get that the degree of y is exactly 1 in the next step. Furthermore, endpoint y is now the only degree-1 node. Hence y is selected and matched next. A contradiction, since y is a path endpoint. \square

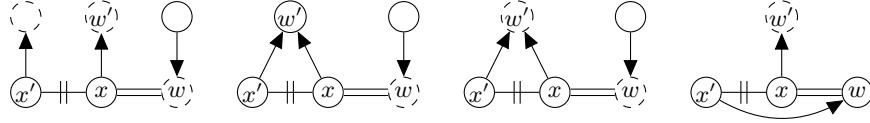


Fig. 5. Removing the last M^* -edge at an end of a path: dashed endpoints have degree one after x and x' are matched (not all edges are drawn)

4 Maximum Degree $\Delta \geq 4$

In this section we show

Theorem 2. *The 1-2-MINGREEDY algorithm achieves approximation ratio at least $\frac{\Delta-1}{2\Delta-3}$ for graphs of maximum degree $\Delta \geq 4$.*

In the proof we have to verify bounds ($-$ Edge), ($+$ Endpoints), (\pm Singleton), and (\pm Path). First, we recall these bounds in an overview of how we refine the analysis applied for $\Delta = 3$. In particular, the overview is structured so as to highlight the two main concepts used in our analysis.

- Canceling Transfers

Since Δ is now larger than 3, a path endpoint might receive more credits than for $\Delta = 3$. However, extra transfers degrade the balance of their source components.

To annihilate this drawback we *cancel* extra transfers: certain transfers as per Definition 1 will no longer move coins. The discussion is found in Section 4.1.

In particular, we carefully cancel transfers such that Lemma 5 still applies: each endpoint of a path X still receives at least one credit. Consequently, bound (+Endpoints) of at least

$$c_X \geq 2 \quad (+\text{Endpoints})$$

credits to X will be satisfied.

– Donations

We discuss how to make a path X balanced. Assuming that (+Endpoints) and (−Edge) hold, i.e. that the minimum balance of X is $c_X - d_X \geq 2 - D_X$, observe that

$$c_X - d_X \geq 2 - D_X + 2(\Delta - 2) \quad (\pm\text{Path})$$

holds if X has at least $2(\Delta - 2)$ missing debits. However, we will see that the number of missing debits from X might smaller, even as small as a constant. To increase the balance of X we introduce a *donation* to X , which will compensate excessively payed debits from X . Summing up, path X will have to pay $2(\Delta - 2)$ debits less than maximum, as desired. Like transfers, donations are edges in F . The discussion is found in Section 4.2.

The balance of the source component Y of a donation is reduced, since coins have to be payed. Is Y balanced? If Y is a singleton, then we show that both nodes of Y pay at most

$$d_Y \leq 2(\Delta - 2) \quad (\pm\text{Singleton})$$

coins even if a donation must be payed. For a path Y we have to verify (−Edge) for each M -edge $\{y, y'\}$ of Y , i.e. we have to show that nodes y and y' pay at most

$$2(\Delta - 2) \quad (-\text{Edge})$$

coins even if a donation must be payed.

Organization of the Proof. In Section 4.1 we introduce transfer cancellations and verify bound (+Endpoints). Thereafter we discuss donations in Section 4.2. In Section 4.3 we develop bounds on the number of coins payed by edges with an outgoing donation. Finally, in Section 4.4 we show that all components are balanced by proving (−Edge), (\pm Singleton), and (\pm Path).

4.1 Canceling Extra Transfers

Unlike for $\Delta = 3$ where each path endpoint has at most two incident F -edges and therefore receives at most two credits, for $\Delta \geq 4$ we will see that a path endpoint might receive up to three credits. But extra credits decrease the balances of their source components. In order to push up their balances, we show how to cancel extra credits in Definition 2. Thereafter we verify (+Endpoints) in Lemma 9.

Example. We prepare an example of an endpoint w which receives an extra credit. Assume that w receives k credits from nodes matched in steps $1, \dots, s$: in any step $s + l$ with $l \geq 1$ we say that w *already receives* k credits.

Consider Figure 6, where 1-2-MINGREEDY begins by creating the $\Delta - 5$ many $\frac{1}{2}$ -paths drawn above path X , then proceeds to pick the two edges of X in steps $\Delta - 4$ and $\Delta - 3$, and eventually picks the two singletons drawn left and right of X . Endpoint w receives three credits in total: in step $\Delta - 3$ node w already receives two credits from the nodes matched to create path X , and afterwards w receives an additional credit.

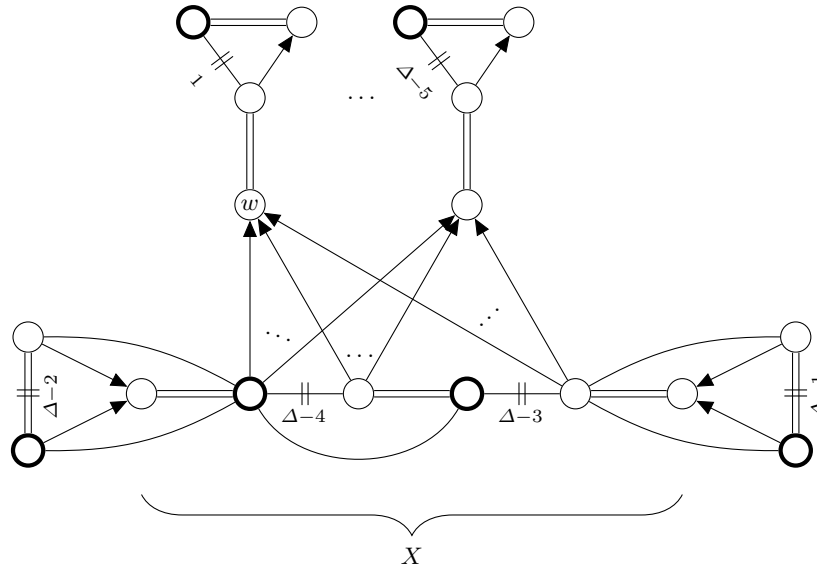


Fig. 6. Endpoint w receives three credits from path X (M -edges are numbered to indicate the order in which they are picked, where 1-2-MINGREEDY selects fat nodes)

However, path X is not balanced. Why? Path X pays three debits to *each* of the $\Delta - 5$ topmost $\frac{1}{2}$ -paths, and receives only four credits from the two singletons. Hence the balance of X is as small as $c_X - d_X = 4 - 3(\Delta - 5)$, whereas balance at least $c_X - d_X \geq 2 - 2m_X(\Delta - 2) + 2(\Delta - 2) = 2 - 2(\Delta - 2)$ is required by $(\pm\text{Path})$.

The Definition. Therefore we cancel an extra credit to w , thereby limiting the number of credits to w to at most two (like for $\Delta = 3$). To prepare the definition we first analyze the exact “configuration” of a path endpoint with more than two credits.

Lemma 8. *As a consequence of Definition 1 of transfers,*

- a) each path endpoint receives at most three credits, and*
- b) path endpoint w receives three credits if and only if the the following holds:*
 - i. from $d(w) = 3$ to $d(w) = 1$ when two incident F -edges are removed and*
 - ii. from $d(w) = 1$ to $d(w) = 0$ when the last incident F -edge is removed.*

Proof. First we argue that an endpoint w receives *less* than three credits if there is a step s when the current degree of w is $d(w) = 2$. By Definition 1, no F -edge of w removed before step s is a credit to w . Moreover, in step s at most two F -edges are still incident with w .

Consequently, if endpoint w receives at least three credits, then w never has current degree $d(w) = 2$. This implies the following two facts. First, the degree of w in G is $d_G(w) \geq 3$, since the degree of w in G is $d_G(w) \geq 2$ by Lemma 2. Secondly, there is a step when the current degree of w is $d(w) = 3$: the degree $d(w)$ of w never equals two, hence we get that $d(w)$ drops from larger than $d(w) > 2$ to below $d(w) < 2$, which is only possible if $d(w)$ drops from exactly $d(w) = 3$, since in each step at most two edges incident with w are removed.

a) In the step when $d(w) = 3$ holds, by Definition 1 of transfers endpoint w still has zero credits. Hence only the remaining three incident edges can be transfer F -edges. Thus w receives at most 3 credits.

b) In the step when $d(w) = 3$ holds, all remaining edges incident with w must be F -edges, since otherwise w would receive less than three credits. The statement follows, since we have already argued that w never has degree $d(w)=2$. \square

If endpoint w receives the maximum of three credits, then we cancel the “third” credit, i.e. the one coming from a node getting matched in the step when the degree of w drops from one to zero, cf. Lemma 8 b) ii. (Observe that path X in Figure 6 is now balanced, since the debits from its rightmost M -covered node are canceled and therefore the balance $c_X - d_X \geq 4 - 2(\Delta - 5) > 2 - 2(\Delta - 2)$ is now strictly larger than $(\pm \text{Path})$ requires.)

Definition 2. *Let w be a path endpoint with current degree $d(w) = 1$ and one incident F -edge $\{v, w\}$. If w already receives two credits, then we cancel transfer (v, w) , i.e. edge $\{v, w\}$ does not move coins.*

The final set of transfers is given by Definitions 1 and 2. We are now ready to prove (+Endpoints).

- Lemma 9.** *a) Each path endpoint w receives at most two credits. An endpoint for which a credit was canceled receives exactly two credits.*
- b) Each path endpoint w receives at least one credit. Therefore (+Endpoints) holds.*

Proof. a) is a direct consequence of Lemma 8 and Definition 2, since w receives at most three credits, and if so then the third credit is canceled. To prove b) we only have to consider path endpoints with less than two credits. In particular, we only have to study an endpoint w for which no credit was canceled, since by a) only such an endpoint can have less than two credits. Now observe that Lemma 5 applies to w , no matter if a credit was canceled for w or not: by Lemma 5 endpoint w receives at least one credit. \square

4.2 Moving Coins to Paths in Donations

We prepare the discussion and introduce some notation, which will also be used in the remainder of Section 4. Let X be a path and consider the creation step of X . We denote the current degree of a node x as $d(x)$. The nodes matched in the creation step are called u and v , where we assume that 1-2-MINGREEDY selects u and matches u with neighbor v . In the next step, we denote the current degree of x as $d'(x)$, and we call the matched nodes u' and v' , where we assume that u' is selected and matched with neighbor v' .

Organization of this Section. We begin this section with an example of a path which pays many debits from nodes u and v . Thereafter, we sketch that nodes u' and v' are good candidates to compensate for the excessively payed debits from u and v . We conclude with the definition of donations—which move coins over F -edges—and a discussion of *donation steps*, i.e. the steps matching source nodes of donations.

Example. Recall that to verify $(\pm\text{Path})$ we would like to show that for each path at least $2(\Delta - 2)$ debits are missing. However, even after canceling transfers a path might pay too many debits. In particular, the example in Figure 6 can be changed slightly as follows, see Figure 7: like in Figure 6 the 1-2-MINGREEDY algorithm first creates the topmost $\Delta - 5$ many $\frac{1}{2}$ -path and creates the left and right singletons in the last two steps; however, in steps $\Delta - 4$ and $\Delta - 3$ the algorithm creates a $\frac{1}{2}$ -path X and another singleton. Path X receives $c_X = 4$ credits and pays $d_X = 2(\Delta - 5)$ debits, i.e. its balance is $c_X - d_X = 4 - 2(\Delta - 5)$, whereas balance at least $c_X - d_X \geq 2 - 2m_X(\Delta - 2) + 2(\Delta - 2) = 2$ is required by $(\pm\text{Path})$.

Sketch. Hence we have to provide more coins to an unbalanced path X . Therefore we focus on the creation step of X as well as on the following step. Recall that $(\pm\text{Path})$ holds if nodes u and v pay $k = 0$ debits, since then at least $2(\Delta - 2)$ debits are missing in total. If nodes u and v pay

$$k > 0$$

coins over debits, then it will turn out that u' and v' have missing debits, which can be used to compensate for excessively payed debits from u and v . Depending on whether u' and v' belong to X or not we proceed as follows:

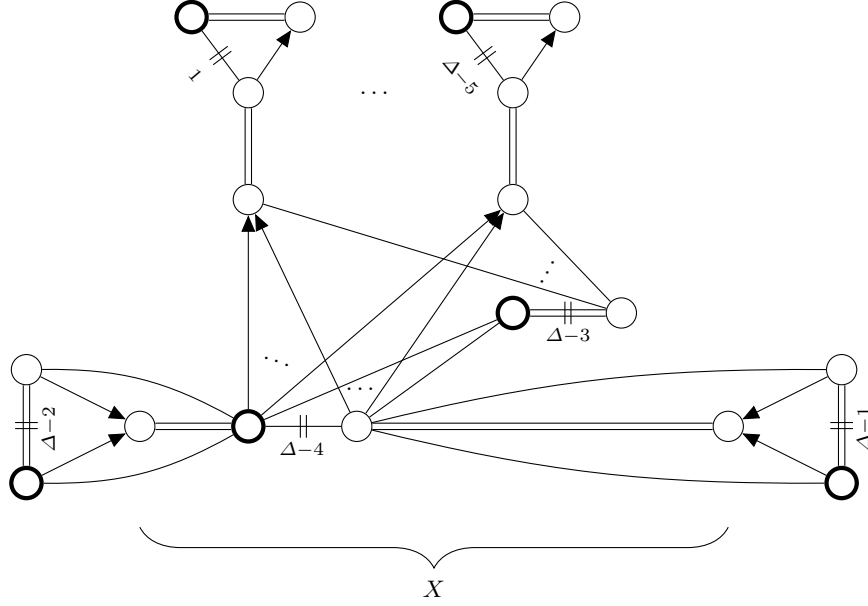


Fig. 7. A $\frac{1}{2}$ -path X which pays too many debits (M -edges are numbered to indicate the order in which they are picked, where 1-2-MINGREEDY selects fat nodes)

- If u' and v' belong to X , then the numbers of missing debits for u, v, u' and v' will add up to at least $2(\Delta - 2)$: the balance of X is $c_X - d_X \geq 2 - D_X + 2(\Delta - 2)$, as required in $(\pm\text{Path})$.
- If otherwise u' and v' belong to another component $Y \neq X$, then without Y becoming unbalanced component Y will be able to *donate* k coins to u and v : the k debits paid by u and v are compensated and again the balance of X is increased by at least $2(\Delta - 2)$ over $2 - D_X$.

We call the step in which u' and v' are matched the *donation step* of X .

Degree-1 Endpoints After Creation. We argue that debits are missing for u' and v' . Recall that we need only consider the case that a debit leaves u or v , say to endpoint w , since otherwise we have found sufficiently many missing debits for $(\pm\text{Path})$ to hold. Definition 1 of transfers requires that w has degree at most $d'(w) \leq 1$ after the creation step of X . The key to our proof is:

endpoint w might have degree *exactly* $d'(w) = 1$ after creation of X .

Definition 3. Let X be a path being created when 1-2-MINGREEDY selects node u and matches it with v . Assume that u or v pays a debit, say to endpoint w . If after creation of X endpoint w has degree *exactly* $d'(w) = 1$, then we say that a degree-1 endpoint exists after creation of X .

Assume that a degree-1 endpoint exists after creation of X , call it w . Since endpoint w will never be matched, the 1-2-MINGREEDY algorithm selects another degree-1 node, namely node u' , next. As desired, we have found missing debits for u' : since u' has degree $d'(u') = 1$ when being selected, node u' pays zero debits by Definition 1.

In Lemma 10 we identify the “configurations” in which a degree-1 endpoint exists after creation of a path. In particular, in Lemma 10 b) we identify the exact configurations in which *no* degree-1 endpoint exists after creation.

Lemma 10. *Consider the creation step of path X , where nodes u and v are matched and 1-2-MINGREEDY selects u with current degree $d(u)$. Assume that one of u or v pays a debit t , say to node w .*

- a) *If $d(u) \geq 3$ holds, then a degree-1 endpoint exists after creation of X .*
- b) *If $d(u) = 2$ holds, then a degree-1 endpoint exists after creation of X , unless the following holds (see Figure 8 for an illustration of this exception):*
 - i. *after creation of X endpoint w has degree $d'(w) = 0$,*
 - ii. *at creation of X endpoint w has degree $d(w) = 2$,*
 - iii. *no debit leaves u ,*
 - iv. *debit t leaves v , i.e. we have $t = (v, w)$,*
 - v. *we have $\{u, w\} \in M^*$,*
 - vi. *at creation of X all other endpoints w_1, \dots, w_k neighboring v (i.e. we have $w \notin \{w_1, \dots, w_k\}$) have degree at least three, and*
 - vii. *node v pays no other debits besides t .*

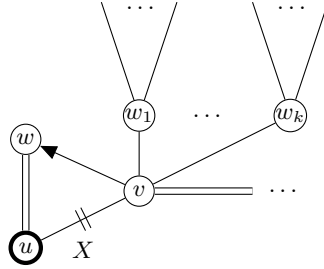


Fig. 8. No degree-1 endpoint exists after creation of path X (the M^* -neighbor of v could be one of the w_i)

Proof. By Definition 1, endpoint w has degree at most $d'(w) \leq 1$ in the step after creation of X . Observe that before creation of X endpoint w is not isolated, since t is an incoming transfer and hence edge t is still incident with w .

a) If $d(u) \geq 3$ holds, then the degree of w drops from at least $d(w) \geq 3$ to at least $d'(w) \geq 1$, since at most two edges incident with w are removed. Hence w has degree exactly $d'(w) = 1$ after creation of X .

b) Now assume that $d(u) = 2$ holds. Since we have $d'(w) \leq 1$ after creation of X , to prove the statement we only need to study the case that no degree-1 endpoint exists after creation, i.e. that endpoint w has degree $d'(w) = 0$ after creation. Observe that i. holds and it remains to verify ii. to vii.

- ii. Endpoint w has degree $d(w) = 2$ at creation, since the degree is $d(w) \geq 2$ by Lemma 2 and if the degree was at least $d(w) \geq 3$ then it could not drop to $d'(w) = 0$ after creation, as is fact by i.
- iii. Since u has degree $d(u) = 2$ at creation of X , node u is not incident with an F -edge, as would be required in order to pay a debit.
- iv. Since debit t does not leave u by iii., debit t leaves v .
- v. Since in the creation step of X the degree of w drops by two, an edge connecting w with each of u and v is removed, and one of both edges must be the M^* -edge incident with w . But we have $t = (v, w)$, i.e. edge $\{v, w\}$ is an F -edge. We obtain $\{u, w\} \in M^*$.
- vi. Observe that for each w_i we have $d'(w_i) \geq 1$ after creation of X , since before creation of X we have $d(w_i) \geq d(u) = 2$ and w_i is not connected with u . Since we assumed that no degree-1 endpoint exists after creation of X and w_i is not isolated then, we obtain $d'(w_i) \geq 2$. But an edge incident with w_i is removed in the creation step of X . Hence we get $d(w_i) \geq 3$.
- vii. This follows from vi. Why? If all w_i have degree at least three at creation of X , then no w_i receives a transfer from v , since the degree of each w_i drops by exactly one (recall that u is adjacent only with v and w). Consequently, endpoint w receives the only transfer from v . \square

The Definition of Donations. Donations will only be needed to move coins to paths for which a degree-1 endpoint exists after creation. Hence in the rest of this section we focus on this case. (In Lemma 18 we analyze the case that no degree-1 endpoint exists after creation.)

Assume that a degree-1 endpoint exists after creation of path X . Since u' is selected with current degree $d'(u') = 1$ and hence pays zero debits, node u' is a preferred candidate to compensate the debits payed by u and v .

To prepare the definition of donations, in Lemma 11 we show that an edge e connecting u' with u or v is removed in the creation step of X . In particular, if a donation needs to be given to X from another component $Y \neq X$, then e is an edge in F .

Lemma 11. *Assume that a degree-1 endpoint exists after creation of path X . The node u' selected next by 1-2-MINGREEDY is*

- a) *an M^* -neighbor of u or v in X or*
- b) *an F -neighbor of u or v in a component $Y \neq X$.*

Proof. After creation of X let w be a degree-1 endpoint. Since at creation of X all degrees are at least two, thereafter both degree-1 nodes w and u' lost an edge connecting to u or v . Hence node u' was connected to u or v by an M^* -edge or by an F -edge.

If u' was connected by an M^* -edge, then node u' belongs to X . This proves a).

To prove b), assume that u' is not an M^* -neighbor of u or v . Then u' was connected only by an F -edge with u or v . It remains to show that u' is not a node of X . To see this, observe that after creation of X all M -covered nodes of X (other than the M^* -neighbors of u and v) have degree at least two and recall that u' is selected when it has degree $d'(u') = 1$. \square

Definition 4. Let nodes u and v be matched in the creation step of path X , where 1-2-MINGREEDY selects node u with current degree $d(u)$. Assume that u and v pay $k > 0$ debits. Let nodes u' and v' of component $Y \neq X$ be matched in the donation step of X , where 1-2-MINGREEDY selects u' with current degree $d'(u')=1$. Denote the F -edge connecting u' with u or v as $\{u', x\}$ for $x \in \{u, v\}$.

- If $d(u) = 2$ holds, then edge $\{u', x\} = \{u', v\}$ is called a static donation and moves $\Delta-3$ coins to v .
- If $d(u) \geq 3$ holds, then edge $\{u', x\}$ is called a dynamic donation and moves k coins to x .

Donation $\{u', x\}$ is denoted as a bold directed edge $(\mathbf{u'}, \mathbf{x})$ from the paying node u' of Y to the receiving node x of X .

Soundness of the Definition. Recall that our proof of $(\pm\text{Path})$ crucially relies on bound $(-\text{Edge})$, i.e. on the (yet unproven) fact that each M -edge of path X pays at most $2(\Delta-2)$ coins over outgoing debits and donations: in total, all M -edges of X pay at most $D_X = m_X \cdot 2(\Delta-2)$ coins. Since the endpoints of X receive at least $c_X \geq 2$ credits by $(+\text{Endpoints})$, the balance of X is at least $c_X - d_X \geq 2 - D_X$. Our approach to use a donation to compensate *all* coins paid by nodes u and v implies that the balance of X is increased by at least $2(\Delta-2)$: the balance is at least $c_X - d_X \geq 2 - D_X + 2(\Delta-2)$, as required in $(\pm\text{Path})$.

But to develop our Definition 4 of donations we only considered coins paid by u and v over debits: if u or v pays a donation whose coins are *not* compensated, then our approach to verify $(\pm\text{Path})$ fails. However, Lemma 12 a) implies that neither u nor v pays a donation, i.e. our Definition 4 of donations is consistent with our approach to analyze the creation step of X . Moreover, Lemma 12 b) shows that coins donated to X need not be shared with another path.

Lemma 12. Let set $C = \{s_1, s_2, \dots\}$ with $s_1 < s_2 < \dots$ be the set of path creation steps, and set $D = \{s_1+1, s_2+1, \dots\}$ the set of donation steps.

- a) We have $C \cap D = \emptyset$.
- b) The node selected in donation step s_i+1 donates coins to exactly one node, which was matched in the creation step s_i .

Proof. a) Consider an arbitrary donation step s_i+1 . Assume that 1-2-MINGREEDY selects node u'_i , and recall that u'_i is selected with current degree $d'(u'_i) = 1$. However, Lemma 2 shows that in each path creation step s_j a node u_j of current degree at least $d(u_j) \geq 2$ is selected. We get $s_i+1 \notin C$ and the statement follows.

b) It suffices to show that $s_{i-1}+1 < s_i$ holds, since then creation step s_i is the only creation step between donation step s_i+1 and the previous donation step $s_{i-1}+1$. The previous donation step $s_{i-1}+1$ selects a node of degree one, whereas step s_i selects a node of degree at least two in order to create a path. Therefore $s_{i-1}+1 \neq s_i$ holds. Using $s_{i-1} < s_i$ we obtain $s_{i-1}+1 < s_i$. \square

4.3 Combined Coins of Transfers and Donations

In this section we prove the following result, which will be applied in Section 4.4 to verify bounds $(\pm\text{Path})$, $(\pm\text{Singleton})$, and $(-\text{Edge})$.

Lemma 13. *Let X be a path for which a degree-1 endpoint exists after creation, and assume that nodes u and v are matched in the creation step of X and nodes u' and v' are matched in the next step.*

Denote by $d_{u,v}$ the number of debits payed by u and v , and by $d_{v'}$ the numbers of debits payed by v' (recall that u' pays no debits since u' is selected with current degree one). There exist (non-negative) integers k and l with

$$d_{u,v} \leq k \quad \text{and} \quad d_{v'} \leq l$$

such that

$$k + l \leq 2(\Delta - 2)$$

holds. In particular, if u' belongs to another component $Y \neq X$ and pays a donation to X , then we may choose k as the number of donated coins.

Organization of the Proof. To verify Lemma 13 we distinguish the following cases. Assume that path X is created when u is selected with current degree $d(u)=2$. Then we bound $k+l$ whether u' is a node of X or of another component $Y \neq X$, where by Definition 4 the analysis of the latter case involves the use of a static donation. Assuming that path X is created when u is selected with current degree $d(u) \geq 3$, then again we bound $k+l$ whether u' belongs to X or not, where by Definition 4 in the latter case we have to take into account a dynamic donation. Cases $d(u) = 2$ and $d(u) \geq 3$ are analyzed in Lemma 14 resp. Lemma 15.

In the proof of Lemma 14 it suffices to analyze (the number of debits leaving) each of nodes u, v, u' , and v' individually. However, in order to show Lemma 15 we have to take into account that the bound l depends “dynamically” on the number k of debits payed by u and v . Consequently, the proof is more involved than that of Lemma 14. In particular, we study the endpoints neighboring nodes u and v as well as nodes u' and v' (in G). Therefore we introduce some helpful notation in Definition 5.

Lemma 14. *Lemma 13 holds in case 1-2-MINGREEDY selects u with current degree $d(u) = 2$. In particular, we have the following:*

- a) *If u' is an M^* -neighbor of u or v , i.e. node u' belongs to X , then we may choose k and l such that we have $d_{u,v} = k \leq \Delta - 2$ as well as $d_{v'} = l \leq \Delta - 2$.*
- b) *If u' belongs to component $Y \neq X$, then we may choose k and l such that we have $d_{u,v} \leq k = \Delta - 3$ as well as $d_{v'} = l \leq \Delta - 1$.*

In both cases we have $k + l \leq 2(\Delta - 2)$.

Proof. To prepare the proof, consider the creation step of X . Observe that u is not incident with an F -edge, since we have $d(u) = 2$. Hence u pays zero debits. Recall that also u' pays no debits since 1-2-MINGREEDY selects u' with current degree $d'(u') = 1$.

We prove a). At creation of X node v is incident with at most $\Delta - 2$ edges of F , since v is also incident with its M -edge and M^* -edge. Hence v pays at most $\Delta - 2$ debits. We obtain $d_{u,v} \leq \Delta - 2$, since u pays no debits. Analogously, after creation of X node v' is incident with at most $\Delta - 2$ edges of F , since v' is also a node of path X . Here we get $d_{v'} \leq \Delta - 2$. Now choose $k = d_{u,v}$ and $l = d_{v'}$.

We prove b). Since u' does not belong to X and is selected with degree $d'(u') = 1$, an F -edge e incident with u' is removed in the creation step of X . Edge e connects u' with v , since u is not incident with any F -edges when being selected by 1-2-MINGREEDY. In particular, edge e is a static donation from u' to v by Definition 4. But both u' and v are M -covered, thus e is not a debit from v and at most $\Delta - 3$ debits leave v . We get $d_{u,v} \leq \Delta - 3$. Now observe that the M -edge incident with v' is not a debit from v' . Hence we have $d_{v'} \leq \Delta - 1$. Now choose k as the number of donated coins $k = \Delta - 3$ as well as $l = d_{v'}$. \square

Definition 5. *Consider the creation step of path X , when 1-2-MINGREEDY selects node u with current degree $d(u) \geq 3$ and matches u with v . We denote*

- *by \mathcal{W} the set of path endpoints w with current degree at least $d(w) \geq 3$,*
- *by $W \subseteq \mathcal{W}$ the set of \mathcal{W} -endpoints neighboring u or v ,*
- *by $W_\delta = \{w \in W : d'(w) = \delta\}$ the set of W -endpoints which have degree δ after creation of X for $\delta \in \{1, 2\}$,*
- *by $W_{\geq 3} = W \setminus (W_1 \cup W_2)$ the set of W -endpoints which have degree at least 3 after creation of X ,*
- *by $W_1^f = \{w \in W_1 : |\{\{w, u\}, \{w, v\}\} \cap F| = f\}$ the set of W_1 -endpoints connected to u and v by f edges of F for $f \in \{1, 2\}$, and*
- *by $\mathcal{E}(W) = \{\{x, y\} \in E : x \in \{u, v\}, y \in W\}$ the set of edges connecting nodes u and v with W -endpoints.*

Note that W_1^1 and W_1^2 form a partition of W_1 and that W_1 , W_2 , and $W_{\geq 3}$ form a partition of W , i.e. sets $\mathcal{W} \setminus W$, $W_{\geq 3}$, W_2 , W_1^2 , and W_1^1 are pairwise disjoint.

Lemma 15. *Lemma 13 holds in case 1-2-MINGREEDY selects u with current degree $d(u) \geq 3$. In particular, we may choose k and l such that we have*

$$d_{u,v} = k = 2|W_1^2| + |W_1^1| \quad \text{and} \quad (a)$$

$$d_{v'} = l \leq |W_1^1| + |W_2| \quad (b)$$

as well as

$$\begin{aligned} k + l &\leq 2|W_1^2| + 2|W_1^1| + |W_2| \\ &\leq |\mathcal{E}(W)| \end{aligned} \tag{c}$$

$$\leq 2(\Delta - 2). \tag{d}$$

Proof. We prove (a). Therefore we show that nodes u and v pay $d_{u,v} = 2|W_1^2| + |W_1^1|$ debits. To prove that we may choose $k = d_{u,v}$ we have to verify that if X receives a donation then this donation moves $d_{u,v}$ coins. By Definition 4, path X receives a dynamic donation since u was selected with current degree $d(u) \geq 3$. Now observe that a dynamic donation moves exactly $d_{u,v}$ coins.

To count the number of debits paid by u and v , we first study which endpoints do *not* receive a transfer from u or v . First, recall that each endpoint $w \in \mathcal{W}$ has degree at least three before creation of X and observe that w has degree at least one thereafter, since degrees drop by at most two when X is created. Endpoints in $\mathcal{W} \setminus W$, i.e. endpoints which are not adjacent to u or v , as well as endpoints in $W_{\geq 3}$ still have degree at least three after creation of X and hence do not receive credits from u or v . Also, endpoints in W_2 receive no credits from u or v , since their degree is at least two after creation.

Hence only endpoints in W_1 might receive credits from u or v , since their degree is exactly one after creation. In the creation step of X , for each $w \in W_1$, either two incident F -edges are removed, i.e. we have $w \in W_1^2$, or one incident F -edge and an incident M^* -edge is removed, i.e. we have $w \in W_1^1$: since all these removed F -edges are transfers, we get that nodes u and v pay exactly $d_{u,v} = 2|W_1^2| + |W_1^1|$ debits.

To show (b), we prove that node v' pays at most $d_{v'} \leq |W_1^1| + |W_2|$ debits and set $l = d_{v'}$. Which endpoints do *not* receive transfers from v' ? (Recall that u' pays no debits since u' is selected when it has degree $d'(u') = 1$.) Here, our Definition 2 of transfer cancellations is crucial: observe that v' does not pay transfers to endpoints in W_1^2 , since each of these endpoints already receives two credits from u and v and any additional credit from v' is canceled by Definition 2. Furthermore, endpoints in $\mathcal{W} \setminus W$ have degree at least three after creation of X and do not receive transfers from v' : when u' and v' are matched, node u' is selected with current degree $d'(u') = 1$, thus the degree of each endpoint in $\mathcal{W} \setminus W$ drops by at most one and to at least two. Analogously, no endpoint in $W_{\geq 3}$ receives a transfer from v' .

Hence v' might pay debits only to endpoints in W_1^1 and W_2 , and (b) follows. We note that, in particular, an endpoint $w \in W_1^1 \cup W_2$ receives *less* than two credits from u or v , i.e. further credits are not canceled, and we have $d'(w) \leq 2$, i.e. the degree of w might drop to at most one when edge $\{v', w\}$ is removed.

We prove (c), i.e. that $k + l$ is bounded from above by the number $|\mathcal{E}(W)|$ of edges connecting u or v with an endpoint before creation of X . First, we argue that

$$|W_2| \leq |\mathcal{E}(W)| - 2(|W_1^1| + |W_1^2|)$$

holds: for each $w \in W_1^1$ as well as for each $w \in W_1^2$ two edges connecting w with u and v are removed when in the creation step of X the degree of w drops from at least $d(w) \geq 3$ to at most $d'(w) \leq 1$; moreover, each endpoint in W_2 is connected with u or v by at least one of the remaining edges. Now we apply (a) and (b) to obtain $k + l \leq 2|W_1^2| + 2|W_1^1| + |W_2| \leq |\mathcal{E}(W)|$, as claimed.

We prove (d), i.e. that $|\mathcal{E}(W)|$ is bound from above by $2(\Delta - 2)$. To see this, observe first that before creation of X two edges incident with u do not connect u with an endpoint: edge $\{u, v\}$ is an M -edge and edge $\{u, u'\}$ is removed when in the creation step of X the degree of u' drops from at least $d(u') \geq d(u) \geq 3$ to $d'(u') = 1$. Analogously, edges $\{v, u\}$ and $\{v, u'\}$ do not connect v with an endpoint. Consequently, at most $\Delta - 2$ edges connect each of u and v with an endpoint. The statement follows. \square

4.4 All Components Are Balanced

Recall that in order to prove Theorem 2 we have to verify (+Endpoints), (−Edge), (±Path), and (±Singleton). Recall also that in Lemma 9 we have already shown (+Endpoints), i.e. that each path receives at least two credits at its endpoints.

In Lemma 16 we show (±Singleton) for each singleton.

Also in Lemma 16, we show (−Edge) for path M -edges. Thereafter it remains to prove (±Path) for each path. In particular, we prove (±Path) for each path for which a degree-1 endpoint exists after creation as well as for each path for which there does not exist a degree-1 endpoint after creation: these proofs are provided in Lemma 17 resp. Lemma 18.

In the proof of (±Path) we may assume that the nodes u and v matched to create a path X pay at least one debit, since otherwise at least $2(\Delta - 2)$ debits are missing for X and as required in (±Path) the balance of X is at least $c_X - d_X \geq 2 - D_X + 2(\Delta - 2)$, by (+Endpoints) and (−Edge).

We begin with the proof of (−Edge) and (±Singleton).

Lemma 16. *Bounds (−Edge) and (±Singleton) hold, since for each M -edge $\{y, y'\}$ of a path or a singleton, nodes y and y' pay at most $2(\Delta - 2)$ coins.*

Proof. We have to verify that y and y' pay at most $2(\Delta - 2)$ coins whether a donation must be payed or not. Assume that 1-2-MINGREEDY selects node y and matches y with neighbor y' . Hence a donation might be payed from y , but not from y' .

Assume that y pays a donation. The following argument applies whether $\{y, y'\}$ is a path M -edge or a singleton. By Definition 4 of donations, edge $\{y, y'\}$ is picked in the donation step of a path X for which a degree-1 endpoint exists after creation. By Lemma 13, we have $k + l \leq 2(\Delta - 2)$ for the number k of coins donated by y and the number l of debits payed by y' , i.e. both nodes pay at most $2(\Delta - 2)$ coins.

From here on we assume that nodes y and y' pay only debits. If $\{y, y'\}$ is a path M -edge then observe that each of y and y' is incident with at most $\Delta - 2$ edges of F and recall that only F -edges can be transfers. Thus y and y' pay at most $2(\Delta - 2)$ coins.

If $\{y, y'\}$ is the edge of a singleton Y , then each of y and y' has at most $\Delta - 1$ incident F -edges, i.e. we have $d_Y \leq 2(\Delta - 1)$. In order to prove $(\pm \text{Singleton})$ with balance at least $-d_Y \geq -2(\Delta - 2)$ it suffices to show that at least two debits are missing. Assume otherwise, i.e. that y and y' pay at least $2(\Delta - 2) + 1$ debits. Then each of y and y' has degree at least $\Delta - 1$ when $\{y, y'\}$ is picked, since otherwise one of y and y' would have at most $\Delta - 3$ incident F -edges and hence at least two missing debits. But since both y and y' have degree at least $\Delta - 1 \geq 3$ before creation and a debit leaves y or y' , say to endpoint w , before creation endpoint w has degree at least $d(w) \geq 3$ as well and we obtain that $d'(w) = 1$ holds after creation. Hence a node other than w is selected next after creation of Y , call it z . But z had degree at least 3 before creation of Y as well, hence z must be a degree-1 node after creation: two edges connecting y and y' with z are removed from the graph in the creation step of Y . Both edges are not transfers, since y, y' , and z are M -covered. Consequently, each of y and y' has a missing debit, a contradiction. \square

To complete the proof of Theorem 2, it remains to verify $(\pm \text{Path})$.

Lemma 17. *Bound $(\pm \text{Path})$ holds for each path for which a degree-1 path endpoint exists after creation.*

Proof. Let X be a path for which a degree-1 endpoint exists after creation. Since X receives at least $c_X \geq 2$ credits by $(+ \text{Endpoints})$ and each M -edge of X pays at most $2(\Delta - 2)$ coins by $(- \text{Edge})$, to show $(\pm \text{Path})$ we have to prove that the balance of X is increased above $2 - D_X$ by at least $2(\Delta - 2)$.

Let u' be the node selected next after creation of X , and recall that u' is selected with degree $d'(u') = 1$ since there is a degree-1 endpoint after creation of X . By Lemma 11, node u' is the M^* -neighbor of u or v , or u' belongs to another component $Y \neq X$. We distinguish these two cases.

Assume that u' is the M^* -neighbor of u or v , i.e. edge $\{u', v'\}$ is an M -edge of X . Observe that by $(- \text{Edge})$ the number of coins payed by nodes of the X -edges $\{u, v\}$ and $\{u', v'\}$ is bounded from above by $4(\Delta - 2)$. However, letting k be the number of debits payed by nodes u and v , and letting l be the number of debits payed by nodes u' and v' , Lemma 13 shows that $k + l \leq 2(\Delta - 2)$ holds, i.e. edges $\{u, v\}$ and $\{u', v'\}$ pay at least $2(\Delta - 2)$ coins less than maximum. Therefore we obtain $d_X \leq D_X - 2(\Delta - 2)$ and hence the balance of X is at least $c_X - d_X \geq 2 - D_X + 2(\Delta - 2)$.

Now assume that u' belongs to another component $Y \neq X$. Hence X receives a donation from u' . Here, Lemma 13 shows that the number k of coins received by X over the donation satisfies $k \geq d_{u,v}$ for the number $d_{u,v}$ of debits payed by u and v . Since we have $d_{u,v} - k \leq 0$, we obtain $d_X \leq (m_X - 1) \cdot 2(\Delta - 2) + d_{u,v} - k \leq (m_X - 1) \cdot 2(\Delta - 2) = D_X - 2(\Delta - 2)$ and hence the balance of X is at least $c_X - d_X \geq 2 - D_X + 2(\Delta - 2)$ again. \square

No Degree-1 Endpoint Exists After Creation. By \bar{X} we denote path such that after creation of \bar{X} there does not exist a degree-1 endpoint. Assume that 1-2-MINGREEDY selects node \bar{u} to create \bar{X} and matches \bar{u} with \bar{v} . Lemma 10 iii. shows that node \bar{u} pays no debits, Lemma 10 iv. and vii. show that node \bar{v} pays exactly one debit. Therefore we can only bound the number of missing debits from \bar{u} and \bar{v} by at least $2(\Delta - 2) - 1$. Thus the number of coins payed by \bar{X} is at most

$$d_{\bar{X}} \leq D_{\bar{X}} - (2(\Delta - 2) - 1),$$

where $D_{\bar{X}} = m_{\bar{X}} \cdot 2(\Delta - 2)$ is the maximum possible number of coins payed by \bar{X} , since each M -edge of \bar{X} pays at most $2(\Delta - 2)$ coins by $(-Edge)$.

Can we find an additional missing debit? However, unlike in the case that a degree-1 endpoint exists after creation, in order to find an additional missing debit for \bar{X} we cannot rely on the analysis of a degree-1 node matched after creation of \bar{X} . In particular, no coins are donated to \bar{X} .

Therefore, using $(+Endpoints)$ we obtain that the balance of \bar{X} is at least

$$c_{\bar{X}} - d_{\bar{X}} \geq 2 - (D_{\bar{X}} - (2(\Delta - 2) - 1)) = 1 - D_{\bar{X}} + 2(\Delta - 2),$$

which fails to satisfy bound $(\pm Path)$ by one coin. Thus, if we can find an additional credit or an additional missing debit, then \bar{X} is balanced and we are done. Hence the following result completes the proof of Theorem 2.

Lemma 18. *Bound $(\pm Path)$ holds for each path \bar{X} for which there does not exist a degree-1 path endpoint after creation, since \bar{X} receives $c_{\bar{X}} \geq 3$ credits or an M -edge $\{\bar{x}, \bar{x}'\}$ of \bar{X} pays at most $2(\Delta - 2) - 1$ coins, where $\{\bar{x}, \bar{x}'\} \neq \{\bar{u}, \bar{v}\}$ is not the M -edge of \bar{X} picked in the creation step.*

Proof. To prepare the proof, observe that to show $c_{\bar{X}} \geq 3$ it suffices to identify an endpoint of \bar{X} which receives at least two credits, since by Lemma 5 the other endpoint of \bar{X} receives an additional credit.

We distinguish cases by the number $m_{\bar{X}}$ of M -edges of \bar{X} .

Assume that \bar{X} is a $\frac{1}{2}$ -path with $m_{\bar{X}} = 1$. We show that an endpoint of \bar{X} receives two credits, which proves $c_{\bar{X}} \geq 3$. First, we consider the creation step of \bar{X} and argue that in the subsequent step there is an endpoint of \bar{X} which is not isolated. Since after creation of \bar{X} there does not exist a degree-1 endpoint, by Lemma 10 b) the 1-2-MINGREEDY algorithm selects a node \bar{u} of current degree exactly $\bar{d}(\bar{u}) = 2$. Let \bar{v} denote the node matched with \bar{u} and observe that the M^* -neighbor of \bar{v} , call it \bar{w} , has degree at least $\bar{d}(\bar{w}) \geq \bar{d}(\bar{u}) = 2$ as well. Since $m_{\bar{X}} = 1$ holds, node \bar{w} must be an endpoint of \bar{X} , and since we have $\bar{d}(\bar{u}) = 2$ endpoint \bar{w} is not adjacent to \bar{u} . Thus the degree of \bar{w} drops by at most one and to at least $\bar{d}'(\bar{w}) \geq 1$ in the step after creation of \bar{X} .

Moreover, after creation of \bar{X} all non-isolated path endpoints have degree at least two, since there does not exist a degree-1 endpoint. This holds in particular for \bar{w} . But since after creation of \bar{X} endpoint \bar{w} is incident with at least two F -edges, endpoint \bar{w} eventually receives at least two credits, namely over those two F -edges of \bar{w} which are removed last from the graph.

From here on, assume that \bar{X} has $m_{\bar{X}} \geq 2$ edges of M . We again denote the nodes matched to create \bar{X} by \bar{u} and \bar{v} , where we assume that 1-2-MINGREEDY selects \bar{u} . Recall that since no degree-1 endpoint exists after creation of \bar{X} , by Lemma 10 b) an M^* -edge of \bar{X} incident with an endpoint of \bar{X} is removed from the graph.

We consider the step s when for the second time an M^* -edge incident with an endpoint of \bar{X} is removed from the graph. Denote by \bar{x} the selected node and by \bar{x}' the neighbor matched with \bar{x} . Since $m_{\bar{x}} \geq 2$ holds we have $\{\bar{u}, \bar{v}\} \neq \{\bar{x}, \bar{x}'\}$. We distinguish the cases that \bar{x} and \bar{x}' pay only debits, or \bar{x} pays a static or a dynamic donation.

First, assume that \bar{x} and \bar{x}' pay **no donation** but only debits. To show that edge $\{\bar{x}, \bar{x}'\}$ pays at most $2(\Delta - 2) - 1$ debits, assume the opposite. Then each of \bar{x} and \bar{x}' pays $\Delta - 2$ debits, since each is incident with at most $\Delta - 2$ edges of F . Hence each of \bar{x} and \bar{x}' has degree at least $\Delta - 1$ at step s , since each is incident with $\Delta - 2$ edges of F as well as with edge $\{\bar{x}, \bar{x}'\}$. But a debit leaves \bar{x} or \bar{x}' , say to endpoint w , hence by Definition 1 of transfers after step s endpoint w has degree at most one. In particular, endpoint w has degree exactly one after step s , since before step s node \bar{x} has degree at least $\Delta - 1 \geq 3$ and hence endpoint w has degree at least 3 as well.

Thus a node other than w is selected next, call it y . But node y had degree at least 3 before step s as well and since thereafter endpoint w has degree exactly one, node y is selected with current degree exactly one. Consequently, edges $\{\bar{x}, y\}$ and $\{\bar{x}', y\}$ are removed from the graph in step s . Both $\{\bar{x}, y\}$ and $\{\bar{x}', y\}$ are not transfers, since \bar{x}, \bar{x}' , and y are M -covered. Now observe that at most one of $\{\bar{x}, y\}$ and $\{\bar{x}', y\}$ can be an M^* -edge, since otherwise two M^* -edges would be incident with y : at least one of $\{\bar{x}, y\}$ and $\{\bar{x}', y\}$ is an F -edge, say $\{\bar{x}, y\}$. Since edge $\{\bar{x}, y\}$ is not a transfer, there is a debit missing for \bar{x} and edge $\{\bar{x}, \bar{x}'\}$ pays at most $2(\Delta - 2) - 1$ debits, as claimed.

Now assume that \bar{x} pays a **static donation**. By Definition 4, path \bar{X} pays $\Delta - 3$ coins over the donation from \bar{x} . Furthermore, node \bar{x}' pays at most $\Delta - 2$ debits, since at most $\Delta - 2$ many F -edges are incident with \bar{x}' when 1-2-MINGREEDY picks edge $\{\bar{x}, \bar{x}'\}$. Again, the nodes in edge $\{\bar{x}, \bar{x}'\}$ pay at most $2(\Delta - 2) - 1$ coins.

Lastly, assume that \bar{x} pays a **dynamic donation** (\bar{x}, u) of k coins. By Definition 4 of donations, node u belongs to a path $X \neq \bar{X}$ for which a degree-1 endpoint exists after creation, and 1-2-MINGREEDY selects u with current degree $d(u) \geq 3$ when creating X . Edge $\{\bar{x}, \bar{x}'\}$ is picked in the donation step of X when \bar{x} has current degree $d'(\bar{x}) = 1$.

Assume that u is matched with v , and recall the definition of the sets $W, W_1, W_1^1, W_1^2, W_2$, and $W_{\geq 3}$ of endpoints neighboring u and v , see Definition 5. By Lemmas 13 and 15 we have the following: nodes u and v pay $d_{u,v} = k = 2|W_1^2| + |W_1^1|$ debits, node \bar{x} donates k coins to u , node \bar{x}' pays at most $d_{\bar{x}'} = l \leq |W_1^1| + |W_2|$ debits. Hence \bar{x} and \bar{x}' pay at most $k + l \leq 2(\Delta - 2)$ coins in total. If $k + l < 2(\Delta - 2)$ holds then an additional debit is missing from \bar{x} or \bar{x}' and we are done.

So assume from here on that \bar{x} and \bar{x}' pay $k + l = 2(\Delta - 2)$ coins. As a consequence of Lemma 15 (c) and (d) we get

$$k + l = 2|W_1^2| + 2|W_1^1| + |W_2| = |\mathcal{E}(W)| = 2(\Delta - 2).$$

Since $k = 2|W_1^2| + |W_1^1|$ holds, we get that \bar{x}' pays exactly

$$d_{\bar{x}'} = l = |W_1^1| + |W_2|$$

debts. To which endpoints? Node \bar{x}' pays zero debts to endpoints in $\mathcal{W} \setminus W$ or $W_{\geq 3}$: in the step when 1-2-MINGREEDY picks edge $\{\bar{x}, \bar{x}'\}$ these endpoints have degree at least three, and since these endpoints are not adjacent to \bar{x} when \bar{x} is selected with current degree $d'(\bar{x}) = 1$ in the donation step of X , the degrees of these endpoints drop by at most one and to at least two, i.e. these endpoints do not receive transfers from \bar{x}' . Therefore \bar{x}' might only pay debts to endpoints in $W_2 \cup W_1$. But node \bar{x}' also pays no debts to nodes in W_1^2 , since in the step when 1-2-MINGREEDY picks edge $\{\bar{x}, \bar{x}'\}$ each endpoint in W_1^2 already receives two credits from nodes u and v and any further credits from \bar{x}' are canceled. Consequently, node \bar{x}' might pay debts only to nodes in W_2 or W_1^1 . Since sets $\mathcal{W} \setminus W, W_{\geq 3}, W_2, W_1^2$, and W_1^1 are pairwise disjoint and since we have $d_{\bar{x}'} = |W_1^1| + |W_2|$, we obtain that \bar{x}' pays exactly one debit to each endpoint in W_1^1 as well as to each endpoint in W_2 .

In the rest of the proof we proceed as follows. We show that either an endpoint of \bar{X} receives at least two credits, which proves $c_X \geq 3$, or we show a contradiction to our assumption that nodes \bar{x} and \bar{x}' pay $k + l = 2(\Delta - 2)$ coins, which proves that edge $\{\bar{x}, \bar{x}'\}$ pays at most $2(\Delta - 2)$ coins.

Consider the step when 1-2-MINGREEDY picks edge $\{\bar{x}, \bar{x}'\}$. Recall that \bar{x} is selected with current degree $d'(\bar{x}) = 1$, i.e. when the M^* -edge of \bar{x} is already removed from the graph. Recall also that an M^* -edge incident with a path endpoint of \bar{X} is removed in this step. This M^* -edge must be incident with \bar{x}' and we call it edge $\{\bar{x}', \bar{w}\}$.

We conduct a case analysis based on which of W_1^1, W_1^2 , or W_2 endpoint \bar{w} belongs to. (Recall that sets W_1^1 and W_1^2 form a partition of W_1 and sets W_1, W_2 , and $W_{\geq 3}$ form a partition of W .) Path X is created when the nodes u and v being matched have degree at least $d(u) \geq 3$ resp. $d(v) \geq 3$, and after creation of X the degree of \bar{x} is exactly $d'(\bar{x}) = 1$. Thus two edges $\{u, \bar{x}\}$ and $\{v, \bar{x}\}$ are removed from the graph in the creation step of X .

$\bar{w} \in \mathcal{W} \setminus W$ or $\bar{w} \in W_{\geq 3}$: Here we show that \bar{w} receives at least two credits due to its large degree after creation of X .

Since X receives a dynamic donation, by Definition 4 of donations the degrees of u and v are at least three in the creation step of X . Consequently, the degree of \bar{w} is at least $d(\bar{w}) \geq 3$ as well. If we have $\bar{w} \in \mathcal{W} \setminus W$, then \bar{w} is not adjacent to u or v and hence no edges incident with \bar{w} are removed in the creation step of X . Thus the degree of \bar{w} is at least $d'(\bar{w}) \geq 3$ after creation of X . If $\bar{w} \in W_{\geq 3}$

holds, then the degree of \bar{w} is at least $d'(w) \geq 3$ after creation of X by Definition 5.

In the step after creation of X , i.e. in the donation step of X , when 1-2-MINGREEDY picks edge $\{\bar{x}, \bar{x}'\}$, the degree of endpoint \bar{w} drops by at most one, since \bar{w} is not adjacent with node \bar{x} which is selected with current degree $d'(\bar{x}) = 1$.

Consider the step after edge $\{\bar{x}, \bar{x}'\}$ is picked. The degree of \bar{w} is at least two and the M^* -edge of \bar{w} is removed from the graph. Hence \bar{w} is incident with at least two F -edges. Now observe that \bar{w} eventually receives at least two credits, namely over the two of the F -edges of \bar{w} which are removed last from the graph. Consequently, path \bar{X} receives at least $c_{\bar{X}} \geq 3$ credits.

$\bar{w} \in W_1^1$: We show a contradiction to our assumption that $k + l = 2(\Delta - 2)$ holds.

Consider the step when path X is created. Recall by Definition 5 of W_1^1 , that at least two edges incident with \bar{w} are removed, since \bar{w} has degree at least $d(\bar{w}) \geq 3$ before and degree at most $d'(\bar{w}) \leq 1$ afterwards. Also by definition of W_1^1 , only one F -edge incident with \bar{w} is removed, hence the M^* -edge incident with \bar{w} is removed as well. Consequently, endpoint \bar{w} belongs to X .

We obtain a contradiction, since endpoint \bar{w} belongs to path \bar{X} and $\bar{X} \neq X$ holds by definition of the donation (\bar{x}, u) .

$\bar{w} \in W_1^2$: By Definition 5 of set W_1^2 , two F -edges e_1 and e_2 incident with \bar{w} are removed when path X is created. In particular, since after creation of X the degree of \bar{w} is exactly one, edges e_1 and e_2 are credits to \bar{w} and both credits are never canceled. Hence path \bar{X} receives at least $c_{\bar{X}} \geq 3$ credits.

$\bar{w} \in W_2$: Recall that $d_{\bar{x}'} = l = |W_1^1| + |W_2|$ holds and that node \bar{x}' pays a debit to each endpoint in W_1^1 and to each endpoint in W_2 . Since we have $\bar{w} \in W_2$, node \bar{x}' pays a debit to \bar{w} . But only F -edges can be transfers, thus we have $\{\bar{x}', \bar{w}\} \in F$. A contradiction to endpoint \bar{w} being the M^* -neighbor of \bar{x}' , i.e. to $\{\bar{x}', \bar{w}\} \in M^*$. \square

5 Inapproximability Results

MINGREEDY does not exploit knowledge gathered about the input in previous steps: e.g. the neighbors of the selected node u are not remembered in order to “explore” the neighborhood of u later. In a step of MINGREEDY, an arbitrary node of minimum degree, who is located in an unknown place in the graph, is matched to an arbitrary neighbor.

A question arises naturally: are the worst case performance guarantees given above for MINGREEDY optimal, i.e. is there a greedy matching algorithm which always computes larger matchings than proven for MINGREEDY? In particular, a greedy matching algorithm in question may in each step utilize all previously gathered knowledge in very sophisticated node and edge selection routines.

Adaptive priority algorithms [BNR02] in the *vertex model* [DI04] define a large class of deterministic greedy matching algorithms, which we denote as \mathcal{APV} . \mathcal{APV} -algorithms do not have resource constraints and formalize the essential properties of greedy algorithms: to what extent can the input be unveiled in a single step, what are the possible irrevocable decisions for the constructed solution to be done after part of the input is revealed? An \mathcal{APV} -algorithm may gather and process much data about its input instances and deduce knowledge to be used in clever future steps. Therefore \mathcal{APV} -algorithms seem much stronger than MINGREEDY. In particular, \mathcal{APV} contains (the deterministic variants of) many prominent greedy matching algorithms, see Lemma 19.

Nevertheless, we construct graphs with degrees bounded by Δ for which a matching of size at most $\frac{\Delta-1}{2\Delta-3} + o(1)$ times optimal is computed. So our $\frac{2}{3}$ lower bound for $\Delta = 3$ and our $\frac{\Delta-1}{2\Delta-3}$ lower bound for Δ -regular graphs are tight: the very simple MINGREEDY algorithm has optimal worst case performance among \mathcal{APV} -algorithms. For graphs of degree at most Δ our $\frac{\Delta-1/2}{2\Delta-2}$ lower bound shows that MINGREEDY has good worst case performance.

For an \mathcal{APV} algorithm A the input graph is represented as a set of adjacency lists, e.g. $\{\langle u; v, w \rangle, \langle v; u, w \rangle, \langle w; u, v \rangle\}$ is the triangle on u, v, w (where an arbitrarily ordered list of neighbors appears after a semicolon). An \mathcal{APV} -algorithm A has a priori access to the number of nodes and starts with an empty matching. In each step, algorithm A selects a node by specifying a total priority order on all possible adjacency lists. From the given order, algorithm A receives the adjacency list which has highest priority and corresponds to a still non-isolated node u , say $\langle u; v, w, \dots \rangle$. (A node is called *isolated*, if it, or each of its neighbors, is matched.) Lastly, algorithm A selects a non-isolated matching partner for u from the neighbor set $\{v, w, \dots\}$ and then changes to the next step.

Matched nodes are not removed from the adjacency lists in G : Observation 1 shows that if A remembers already matched nodes, then A can submit priority orders on adjacency lists w.r.t. the “reduced” graph, i.e. w.r.t. the set of nodes which are not yet isolated.

Observation 1. *If an \mathcal{APV} -algorithm receives a data item $\langle u; v, w, x, \dots \rangle$, then each neighbor v, w, x, \dots of u is either matched or not isolated.*

Proof. Let $y \in \{v, w, x, \dots\}$ and assume that y is not matched but isolated. Each neighbor of y is matched and thus isolated. Since u is a neighbor of y and the data item of u is received, node u is not isolated. A contradiction. \square

Leaving adjacency lists unchanged increases the power of A : a neighbor of an already matched node v may be requested and hence A is able to explore the neighborhood of v and is not oblivious to the parts of G being processed.

Lemma 19. *The class \mathcal{APV} contains (all deterministic variants of) GREEDY, KARPSIPSER, MRG, MINGREEDY, SHUFFLE, and all vertex iterative algorithms.*

Proof. We have to implement any deterministic variant of one of the given algorithms as an \mathcal{APV} -algorithm.

In a given round of GREEDY, a priority order e_1, e_2, e_3, \dots on edges has to be built adaptively depending on previous computations. This list can be built edge by edge like this: The sub-list e_2, e_3, \dots is built under the assumption that e_1 is not in the graph, the sub-list e_3, \dots is built under the assumption that both e_1, e_2 are not in the graph, etc. After GREEDY adds the highest priority edge in the graph to the solution, a new round starts and a new priority list is built adaptively like above. We have to translate e_1, e_2, e_3, \dots into a priority order on adjacency lists. Therefore edge $e_i = \{u, v\}$ is replaced by a list containing an adjacency list $\langle u; v, w, x, \dots \rangle$ for each possible choice of w, x, \dots , making sure that if e_i is in the graph it is picked before e_{i+1} . If GREEDY receives the adjacency list $\langle u; v, w, x, \dots \rangle$, node u is matched with v .

The argument for the other algorithms is analogous. In each round, a priority list is built recursively under the assumptions that high priority entries are not in the graph, and then translated to a list of adjacency lists.

KARPSIPSER works like GREEDY but prefers an edge incident with a degree-1 node, if such an edge exists. So edges incident with degree-1 nodes are moved to the front of the priority order, i.e. the priority order on adjacency lists starts with adjacency lists $\langle u; v, w, x, \dots \rangle$ for all possible choices of u and v, w, x, \dots where only v is not already matched.

To implement MRG, a priority list on nodes u_1, u_2, \dots has to be translated. Node u_i is replaced by a list of adjacency lists $\langle u_i; v, w, x, \dots \rangle$ for each possible choice of v, w, x, \dots . If MRG receives an adjacency list, say for node u , an arbitrary non-isolated neighbor of u is to be matched with u .

To implement MINGREEDY, a priority list on node degrees $1, 2, \dots$ has to be translated. Degree i is replaced by a list of adjacency lists $\langle u; v_1, \dots, v_i, w, x, \dots \rangle$ for each possible combination of u and $v_1, \dots, v_i, w, x, \dots$ where only v_1, \dots, v_i are not already matched. As for MRG, for a received adjacency list, say for node u , an arbitrary non-isolated neighbor of u is matched with u .

SHUFFLE does not compute priority lists in each round, but a node permutation u_1, u_2, \dots, u_n is computed once at the start, using the number n of nodes in the graph. A node u_i is replaced by a list of adjacency lists $\langle u_i; v, w, x, \dots \rangle$ for each possible choice of v, w, x, \dots . This priority order is used in each round. If SHUFFLE receives an adjacency list for a node u_i , then from the still non-isolated neighbors of u_i the first one in u_1, u_2, \dots, u_n is matched with u_i .

A vertex iterative algorithm [GT12] considers nodes one at a time, and probes each node u for neighbors. The probing for u ends after the first successful probe, say for neighbor v . Then nodes u and v are matched. The probing for u also ends after all possible neighbors have been tested without success. Furthermore, the probing for u ends if the algorithm decides to stop probing for further neighbors of u . Once the probing for u ends, node u is never considered again and u is never probed as the neighbor of any other node considered later. In a round of

of the algorithm, we denote by $(a, b), (a, c), \dots, (a, d), (e, f), (e, g), \dots, (e, h), \dots$ that node a is to be probed for neighbors b, c, \dots, d , then e is to be probed for neighbors f, g, \dots, h , etc. A pair (u, v) is translated to a list of adjacency lists $\langle u; v, w, x, \dots \rangle$ for each possible choice of w, x, \dots . If an adjacency list $\langle u; v, w, x, \dots \rangle$ is received, then node u is matched with v . \square

To prove that no \mathcal{APV} -algorithm can guarantee approximation ratio better than $\frac{\Delta-1}{2\Delta-3} + o(1)$, we first present in Lemma 20 a construction for \mathcal{APV} -algorithms without access to the number of nodes in the graph, and then adapt the construction in Theorem 3 to the full class of \mathcal{APV} -algorithms.

Lemma 20. *Let A be an \mathcal{APV} -algorithm without access to the number of nodes. There is an input graph of degree at most Δ for which A computes a matching of size at most $\frac{\Delta-1}{2\Delta-3}$ times optimal.*

Proof. We describe the construction of a hard input instance G for algorithm A as a game played between A and an adversary B . As A unveils G only bit by bit, adversary B may actually construct G on the fly, thereby reacting to the various moves of A such that G has a much larger matching than the solution of A . Of course, all adjacency lists presented by B during the whole game have to be consistent with the final graph G constructed by B .

The game consists of the *regular game*, which lasts for $s = \Delta - 3$ steps, followed by the *endgame*, which has two steps.

During the regular game, adversary B maintains the following invariant: each node v that is not yet isolated has an adjacency list of one of the following types.

- Type 1: $\langle v; v_1, \dots, v_d \rangle$ where v and v_1, \dots, v_d are *unknown*, i.e. they did not occur in a previously received adjacency list, and $3 \leq d \leq \Delta$.
- Type 2: $\langle v; v_1, v_2 \rangle$ where v and v_1, v_2 are unknown.
- Type 3: $\langle v; v_1, v_2, v_3 \rangle$ where v and v_1, v_2 are unknown and v_3 is *known*, i.e. node v_3 was received by A in a previous adjacency list.

Observe, that all nodes in G have degree at least two.

Consider the very first step of A . Since all nodes are still unknown, all nodes have adjacency lists of type 1 or 2. Hence the invariant holds. Consider step i and assume that the invariant holds. Adversary B presents the highest ranked adjacency list that is of type 1, 2 or 3. Call that adjacency list a_i .

Case 1: $a_i = \langle v; v_1, \dots, v_d \rangle$ is a type-1 adjacency list. Since all nodes in a_i are unknown, we may w.l.o.g. assume that A matches v with v_1 . Adversary B constructs the connected component C depicted in Figure 9 which consists only of nodes of types 1 and 2. All nodes of C are isolated in the next step, hence the invariant is maintained. Observe that within C the maximum matching M^* scores two edges (the double edges $\{v, v_2\}, \{v_1, v_d\}$ in Figure 9) whereas the matching M computed by A scores just one edge (the crossed edge $\{v, v_1\}$).

Since in Case 1 algorithm A requests only unknown nodes, adversary B is able to trick A into unveiling part of G from which A cannot gather knowledge about the rest of G . Can A act smarter? Assume that A has already received

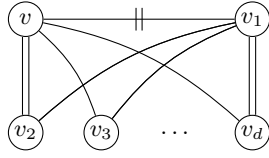


Fig. 9. A connected component of a hard instance

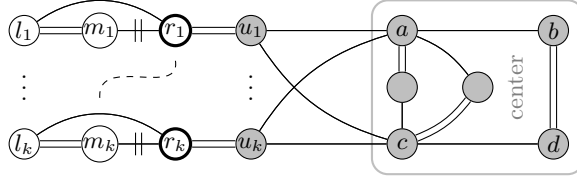


Fig. 10. The core of a hard instance (Gray nodes are unknown, fat frontier nodes. The dashed edge is an example for $\{m_i = v, r_j = v_3\}$ in case 3.)

the adjacency lists of the *middle nodes* m_1, \dots, m_k of the triangles $\{l_j, m_j, r_j\}$ of known nodes connected by *frontier nodes* r_j and unknown nodes u_j to the still unknown *center* of G , see Figure 10. If A requests an unknown node with two unknown neighbors, then B easily tricks A by constructing a new triangle $\{l_i, m_i, r_i\}$.

Case 2: $a_i = \langle v; v_1, v_2 \rangle$ is a type-2 adjacency list. Again, all nodes of a_i are unknown and we may assume that A matches v with v_1 . Adversary B constructs a triangle $\{l_i, m_i, r_i\}$ with $l_i = v_2, m_i = v, r_i = v_1$ and inserts the edge $\{r_i, u_i\}$, with a new unknown node u_i , to connect the triangle to the unknown center. Observe that before nodes m_i, r_i are matched, nodes m_i, l_i are of type 2 and r_i, u_i are of type 1. After matching m_i, r_i , nodes l_i, m_i, r_i are isolated and u_i turns into a type-3 node. Hence the invariant still holds. Again, M^* scores two edges, namely $\{l_i, m_i\}$, $\{r_i, u_i\}$, and M scores the edge $\{m_i, r_i\}$.

Now assume that A tries to explore the neighborhood of known nodes. Observe that the only adjacency lists with a known node are of type 3 and have exactly one unknown node: since the known nodes l_j, m_j, r_j are already isolated, an unknown node can only be explored in the neighborhood of frontier nodes. Again, adversary B tricks A with a new triangle $\{l_i, m_i, r_i\}$.

Case 3: $a_i = \langle v; v_1, v_2, v_3 \rangle$ is a type-3 adjacency list. Since v_3 is known, v_3 occurred in a previously presented adjacency list. Observe that in our construction so far, the only type-3 nodes are unknown neighbors of known frontier nodes. So v is the neighbor of a frontier node $r_j = v_3$ with $j < i$.

Is algorithm successful in exploring the unknown neighbor u_j of r_j , i.e. does $v = u_j$ hold? Not necessarily, since B may on the fly construct further neighbors of r_j . Why? Since r_j gets matched as soon as it becomes known, algorithm A never gets to see the adjacency list of r_j and consequently A can never tell if it already knows all neighbors of r_j . (Adversary B uses this trick here as well as in the end game.)

Since $v_3 = r_j$ is matched and v_1, v_2 are unknown, we may assume that A matches v with v_1 . Adversary B behaves exactly as in case 2 and constructs the triangle $\{l_i = v_2, m_i = v, r_i = v_1\}$ and inserts the edge $\{r_i, u_i\}$ where u_i is a new unknown node. To complete the devious trick, adversary B also inserts the edge $\{m_i = v, r_j = v_3\}$ (see e.g. the dashed edge in Figure 10). Before m_i, r_i are matched, node l_i is of type 2, nodes r_i, u_i are of type 1 and $m_i = v$ is of type

3. After matching m_i, r_i , nodes l_i, m_i, r_i are isolated and u_i turns into a type-3 node. u_j is still of type 3. Hence the invariant still holds. As in case 2, M^* scores $\{l_i, m_i\}, \{r_i, u_i\}$ and M scores $\{m_i, r_i\}$.

This concludes the regular game. In the first step of the endgame adversary B makes algorithm A match a with b . Hence in the next and last step algorithm A matches c . So algorithm A scores two edges in the center, whereas three edges are optimal. As desired, we get

$$|M| = s + 2 = \Delta - 1 \quad \text{and} \quad |M^*| = 2s + 3 = 2\Delta - 3.$$

Observe that our invariant still holds in the first step of the endgame. Again, adversary B presents the highest ranked adjacency list of type 1, 2 or 3. Observe that a and c are the only type-1 nodes left, since the u_j have known neighbors and are of type 3 and all other nodes have degree two and are of type 2. The degree of a and c is $\delta \leq 3 + s$, since both a, c have three center neighbors and each step of the regular game adds at most one neighbor to a respectively c . Let $a_{\Delta-2}$ be the adjacency list received in step $s + 1 = \Delta - 2$.

Case 4a: $a_{\Delta-2} = \langle v; v_1, \dots, v_\delta \rangle$ is a type-1 adjacency list. Since all nodes of $a_{\Delta-2}$ are unknown we may assume that A matches v with v_1 . Adversary B chooses $v = a$, $v_1 = b$ and v_2, \dots, v_δ as the remaining neighbors of a .

Case 4b: $a_{\Delta-2} = \langle v; v_1, v_2 \rangle$ is a type-2 adjacency list. Since all nodes of $a_{\Delta-2}$ are unknown we may assume that A matches v with v_1 . Adversary B sets $v = b$, $v_1 = a$ and $v_2 = d$.

Case 4c: $a_{\Delta-2} = \langle v; v_1, v_2, v_3 \rangle$ is a type-3 adjacency list. As in case 3, the known node v_3 is some matched frontier node $r_j, j < \Delta - 2$ and we may assume that A matches v with v_1 , since v_1, v_2 are unknown. As in case 3, adversary B does *not* present the adjacency list of the unknown node u_j . Instead, B makes b a neighbor of r_j by inserting $\{v_3=r_j, b\}$ —now b has three neighbors—and sets $v=b, v_1=a$ and $v_2=d$.

Adversary B does not violate degree constraints. Nodes introduced in case 1 have degree at most $d \leq \Delta$. All other degrees are at most three, but for a, c and frontier nodes r_j . As discussed, nodes a, c have degree at most $\delta = 3 + s \leq \Delta$. Frontier nodes have degree at most $3 + (s - 1) + 1 = \Delta$, since in each but the first step of the regular game and in step $s + 1$ at most one incident edge is added. \square

Theorem 3. *Let A be an APV-algorithm. There is a graph G of degree at most Δ for which A computes a matching of size at most $\frac{\Delta-1}{2\Delta-3} + \varepsilon$ times optimal for any $\varepsilon > 0$.*

Proof. We modify the adversary B who constructs hard inputs in the proof of Lemma 20. First, the modified adversary B' announces the number $t\Delta$ of nodes, where t is a large integer. Using so many nodes B' constructs $\Omega(t)$ connected components, each with $O(\Delta)$ nodes and approximation ratio no better than $\frac{\Delta-1}{2\Delta-3}$. A negligible portion of the graph might be solved optimally by A . In the proof we frequently refer to the adjacency list types and cases found in the proof of Lemma 20 (types 1, 2, and 3, and cases 1, 2, 3, and 4a-c).

Again, the game between A and B' is split up into the regular game and the endgame. Like the adversary B from the proof of Lemma 20, in each round of the regular game the construction of B' keeps up the invariant that all non-isolated nodes in the graph have adjacency lists of type 1, 2 or 3 and B' returns the highest priority adjacency list having one of these types. However, depending on the requests of A , not only one but several centers C_1, C_2, \dots might be constructed, each in its own connected component of G . Each center C_i is defined as in the proof of Lemma 20, with nodes a_i, b_i, c_i, d_i and two more nodes unique to C_i , see Figure 10. Each C_i will get attached to it a maximum number of triangles, which are not connected to any other center. Thereafter, the nodes of C_i are supposed to be matched in the same order as in the proof of Lemma 20, i.e. when no more triangles are attached, nodes a_i and b_i are matched to each other before c_i is matched. Once c_i is matched, all edges in the connected component of C_i are removed.

Assume that B' has already created the centers C_1, \dots, C_l . Call C_i *active* if a_i is not yet matched with b_i , and *inactive* otherwise. The construction will ensure that C_1, \dots, C_{l-1} are inactive; C_l might still be active. (We note here that after center C_l becomes inactive, there are nodes in the rest of the connected component K of C_l which do *not* have adjacency lists of types 1, 2 or 3. However, all these nodes are neighbors of c_l and adversary B' adds no more nodes to K . Thus A scores at most one more edge in K . Therefore, the additional adjacency list types do not have effect on the rest of the construction and we do not discuss these types explicitly.)

Assume that in the next round A receives a **type-2** adjacency list of a node with two unknown neighbors. Assume that C_l is already inactive, then B' creates the next center C_{l+1} and connects a new type-2 triangle to a_{l+1}, c_{l+1} as described in case 2. If otherwise C_l is still active, let δ be the number of neighbors of a_l, c_l constructed so far and recall that we demand $\delta \leq \Delta$. If $\delta < \Delta$, then B' connects a new type-2 triangle to a_l, c_l as described in case 2. If $\delta = \Delta$, then B' makes A match a_l with b_l as described in case 4b, thereby inactivating C_l .

Assume that in the next round A receives a **type-3** adjacency list. By construction, the received node is unknown and among its three neighbors there is exactly one known node v_3 , where $v_3 = r$ is a frontier node r in the connected component of the still active center C_l . Let δ be the number of neighbors of a_l, c_l constructed so far. If $\delta < \Delta$, then B' connects a new type-3 triangle to a_l, c_l as described in case 3. If $\delta = \Delta$, then B' makes A match a_l with b_l as described in case 4c, and inactivates C_l .

Assume that in the next round A receives a **type-1** adjacency list. If the degree of the received node is smaller than Δ , then B' proceeds as in case 1 and creates a new type-1 connected component. Now assume that the received node has degree Δ . If nodes a_l, c_l have degree less than Δ , then again B' creates a new type-1 connected component. If nodes a_l, c_l have degree $\delta = \Delta$, then B' makes A match a_l with b_l as described in case 4a, and thereby inactivates C_l .

Why is C_l inactivated before B' constructs the next active center? In type-2 and type-3 rounds an increasing number of triangles is connected to C_l until

the degrees of a_l, c_l are Δ . (In intermediate type-1 rounds only type-1 connected components are constructed.) Thereafter, C_l is inactivated in the first type-2 or type-3 round or the first type-1 round in which an adjacency list of a degree- Δ node is received. (In intermediate type-1 rounds with nodes of degree less than Δ only type-1 connected components are constructed.)

The endgame begins as soon as B' has constructed $k \geq t\Delta - 6\Delta$ nodes. Let $\nu = t\Delta - k$ be the number of nodes still to be constructed. Observe that $6\Delta \geq \nu \geq 2\Delta$ holds, since in each round no more than 4Δ additional nodes are introduced (e.g. if $\Delta = 3$ holds and a new triangle is connected to a new center). Since B' has committed to a number of exactly $t\Delta$ nodes, in the first round of the endgame B' utilizes all remaining ν nodes to create additional connected components $\Gamma_1, \dots, \Gamma_c$, each being a complete bipartite graph with 2 nodes on the right side and between 2 and Δ (both 2 and Δ included) nodes on the left. The left sides are as large as possible such that $c = O(1)$ is constant and all nodes in the Γ_i have degree at least two. All nodes in $\Gamma_1, \dots, \Gamma_c$ are still unknown, in particular all nodes have adjacency lists only of types 1 or 2. Since right sides have two nodes, each Γ_i has at most two edges in a maximum matching, making an additional constant number $2c$ of optimal edges in total. We assume that A performs optimally in all Γ_i , thereby scoring $2c$ edges.

Also in the endgame, algorithm A matches still unmatched nodes in the already inactive centers C_1, \dots, C_{l-1} . We assume that A performs optimally also in the connected component of the last center C_l .

What is the approximation ratio of A ? Observe that each type-1 component has $O(\Delta)$ nodes. The same is true for each connected component with a (constant size) center C_i , since $O(\Delta)$ triangles (of constant size) are attached to C_i . So as claimed, $\Omega(t)$ connected components are constructed. Recall that A scores only one out of two edges in each type-1 connected component. On the other hand, in the connected component of an inactive center A achieves approximation ratio exactly $\frac{\Delta-1}{2\Delta-3}$. In all other components, algorithm A even performs optimally. Therefore, to bound the performance of A we may assume that no type-1 components are constructed. Since only C_l might be active at the end of the regular game, there are at least $l-1 = \Omega(t)$ inactive centers, hence the approximation ratio of A is at most

$$\frac{(l-1) \cdot (\Delta-1) + (2\Delta-3) + 2c}{(l-1) \cdot (2\Delta-3) + (2\Delta-3) + 2c},$$

since A performs optimally in C_l and in $\Gamma_1, \dots, \Gamma_c$. Letting $t \rightarrow \infty$ we get $l \rightarrow \infty$ and this ratio is dominated by $\frac{(l-1) \cdot (\Delta-1)}{(l-1) \cdot (2\Delta-3)}$. The statement follows. \square

6 Conclusion

We have analyzed the worst case approximation ratio of the well-known MIN-GREEDY algorithm on graphs of bounded degree. Our performance guarantees of $\frac{2}{3}$ for graphs of degree at most three and of $\frac{\Delta-1}{2\Delta-3}$ for Δ -regular graphs are tight. In particular, MINGREEDY is optimal in the large class of \mathcal{APV} -algorithms,

which contains many prominent greedy matching algorithms. We also proved a performance guarantee of $\frac{\Delta-1/2}{2\Delta-2}$ for graphs of degree at most Δ , and we conjecture that also in this case MINGREEDY is optimal among \mathcal{APV} -algorithms and achieves a worst case approximation ratio of at least $\frac{\Delta-1}{2\Delta-3}$.

Our worst case performance guarantees are stronger than the best known worst case bounds on the expected approximation ratio for the well-known greedy matching algorithms GREEDY, MRG and SHUFFLE, if degrees are small.

Open Questions. Is MINGREEDY optimal among \mathcal{APV} -algorithms on graphs of degree at most Δ ?

What bounds for MINGREEDY can be shown for more restricted graph classes, e.g. bipartite graphs?

Recall that the expected approximation ratio of the randomized MINGREEDY algorithm is $\frac{1}{2} + o(1)$ w.h.p. on graphs of arbitrarily large degree. Does randomized MINGREEDY have an expected approximation ratio strictly better than $\frac{\Delta-1}{2\Delta-3}$ if degrees are bounded by Δ ?

Acknowledgements. I thank Georg Schnitger for many helpful discussions.

References

- ADFS95. Jonathan Aronson, Martin Dyer, Alan Frieze, and Stephen Suen. Randomized greedy matching. ii. *Random Structures & Algorithms*, 6(1):55–73, 1995.
- BNR02. Allan Borodin, Morten N. Nielsen, and Charles Rackoff. (incremental) priority algorithms. In *Proc. 13th SODA*, pages 752–761, 2002.
- CCWZ14. T.-H. Hubert Chan, Fei Chen, Xiaowei Wu, and Zhichao Zhao. Ranking on arbitrary graphs: Rematch via continuous lp with monotone and boundary condition constraints. In Chandra Chekuri, editor, *SODA*, pages 1112–1122. SIAM, 2014.
- DF91. Martin E. Dyer and Alan M. Frieze. Randomized greedy matching. *Random Structures & Algorithms*, 2(1):29–46, 1991.
- DI04. Sashka Davis and Russell Impagliazzo. Models of greedy algorithms for graph problems. In *Proc. 15th SODA*, pages 381–390, 2004.
- FRS93. Alan Frieze, A. J. Radcliffe, and Stephen Suen. Analysis of a simple greedy matching algorithm on random cubic graphs. In *Proc. 4th SODA*, pages 341–351, 1993.
- GT12. Gagan Goel and Pushkar Tripathi. Matching with our eyes closed. In *Proc. 53rd FOCS*, pages 718–727, 2012.
- KS81. Richard M. Karp and Michael Sipser. Maximum matchings in sparse random graphs. In *Proc. 22nd FOCS*, pages 364–375, 1981.
- KVV90. Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proc. 22nd STOC*, pages 352–358, 1990.
- Mag97. Jakob Magun. Greedy matching algorithms, an experimental study. In *Proceedings of the 1st Workshop on Algorithm Engineering*, volume 6, pages 22–31, 1997.

- MP97. Zevi Miller and Dan Pritikin. On randomized greedy matchings. *Random Struct. Algorithms*, 10(3):353–383, 1997.
- MS04. Marcin Mucha and Piotr Sankowski. Maximum matchings via gaussian elimination. In *FOCS*, pages 248–255. IEEE Computer Society, 2004.
- MV80. Silvio Micali and Vijay V. Vazirani. An $O(\sqrt{|V|} |E|)$ algorithm for finding maximum matching in general graphs. In *FOCS*, pages 17–27. IEEE Computer Society, 1980.
- Pol12. Matthias Poloczek. *Greedy algorithms for max sat and maximum matching : their power and limitations*. PhD thesis, Institut für Informatik, Goethe-Universität Frankfurt am Main, 2012. <http://d-nb.info/1036608425>.
- Tin84. G. Tinhofer. A probabilistic analysis of some greedy cardinality matching algorithms. *Annals of Operations Research*, 1(3):239–254, 1984.
- Vaz12. Vijay V. Vazirani. An improved definition of blossoms and a simpler proof of the mv matching algorithm. *CoRR*, abs/1210.4594, 2012.